AD A 077829

HILE COPY

NATIONAL COMMUNICATIONS SYSTEM



TECHNICAL INFORMATION BULLETIN 79-10

MEASUREMENT OF COMPRESSION FACTOR AND ERROR SENSITIVITY FACTOR OF FACSIMILE CODING TECHNIQUES SUBMITTED TO THE CCITT BY GREAT BRITAIN AND THE FEDERAL REPUBLIC

OF GERMANY

OCTOBER 1979

DDC

PEC 10 1979

EUSIVE

APPROVED FOR PUBLIC RELEASE;
DISTRIBUTION UNLIMITED

79 11 27 006

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DDC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

UNCLASSIFIED

BNCS, SEIE

SECURITY CLASSIFICATION OF THIS PAGE (When Date Entered)

REPORT DOCUMEN	TATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORTED	2. GOVY ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
NCS TIB-79-189 111-2-	Communica Factor	THE OF REPORT & PERIOD COVE
A TITLE (and Carried Measurement of and Error Sensitivity Factor		(7)
Techniques Submitted to the		Trinal # /(xt.)
and and fre Feet of f		SORPORMING ON GREPORT NUMBE
the real of	The first of General	1.
Neil/Randally	The second of th	CONTRACT OR GRANT NUMBER(a)
Richard Schaphorst		,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
Steve Urban		DCA100-79-M-0209
9. PERFORMING ORGANIZATION NAME AND	ADDRESS	10. PROGRAM ELEMENT, PROJECT, TA
Delta Information Systems,	Inc. V	AREA & WORK UNIT NUMBERS
259 Wyncote Road		
Jenkintown, PA 19046		
11. CONTROLLING OFFICE NAME AND ADD		13. REPORT-DATE
National Communications Sy Office of Technology and S		Oct eber 19 79/
Washington, D.C. 20305	Standards (NOS-13)	13: NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRES	S(II different from Controlling Office)	15. SECURITY CLASS. (of this report)
	(12) 1161.	WOLACCIETED
	92100	UNCLASSIFIED
	•	154. DECLASSIFICATION/DOWNGRADIN SCHEDULE
16. DISTRIBUTION STATEMENT (of this Repo	ort)	<u> </u>
Distribution unlimited; a	pproved for public relea	ase
	•	
17. DISTRIBUTION STATEMENT (of the about	act entered in Block 20, if different fro	m Report)
19. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if n		Image Statistics
Image Coding Digital Facsimile	Error Sensitivity Two-dimensional Coding	
Facsimile Coding	Coding Algorithms	
Compression Factor	Computer Simulation	
compression rassor	- 3 p	
20. ABSTRACT (Continue en reverse sido II n	scoresty and identify by block number)	
This Technical Information	Bulletin (TIB) describ	es the measurement of
Compression factor and err	•	
		an international standard
by Great Britain and the F		
detailed flow charts and c		
error sensitivity factor a		

TIB is a companion document to NCS TIB 79-9.

SECURITY CLASSIFICATION OF THIS PAGE (When Date Entered)

NCS TECHNICAL INFORMATION BULLETIN 79-10

MEASUREMENT OF COMPRESSION FACTOR

AND ERROR SENSITIVITY FACTOR OF

FACSIMILE CODING TECHNIQUES SUBMITTED

TO THE CCITT BY GREAT BRITAIN AND GERMANY

October 1979

Accession For

NTIS GRA&I
DDC TAB
Unannounced
Justification

By

Distribution/

Availability Codes

Availand/or
special

A S

APPROVED FOR PUBLICATION:

PREPARED BY:

DENNIS BODSON
Senior Electronics Engineer
Office of NCS Technology
and Standards

Marshall L. CAIN

Assistant Manager

Office of NCS Technology
and Standards

FOREWORD

Among the responsibilities assigned to the Office of the Manager, National Communications System, is the management of the Federal Telecommunication Standards Program which is an element of the overall GSA Federal Standardization Program. Under this program, the NCS, with the assistance of the Federal Telecommunication Standards Committee, identifies, develops, and coordinates proposed Federal Standards which either contribute to the interoperability of functionally similar Federal telecommunication systems or to the achievement of a compatible and efficient interface between computer and telecommunication systems. In developing and coordinating these standards a considerable amount of effort is expended in initiating and pursuing joint standards development efforts with appropriate technical committees of the Electronic Industries Association, the American National Standards Institute, the International Organization for Standardization, and the International Telegraph and Telephone Consultative Committee of the International Telecommunication Union. This Technical Information Bulletin presents an overview of an effort which is contributing to the development of compatible Federal, national, and international standards in the area of digital facsimile standards. It has been prepared to inform interested Federal activities of the progress of these efforts. Any comments, inputs or statements of requirements which could assist in the advancement of this work are welcome and should be addressed to:

> Office of the Manager National Communications System ATTN: NCS-TS Washington, D.C. 20305 (202) 692-2124

MEASUREMENT OF COMPRESSION FACTOR

AND ERROR SENSITIVITY FACTOR OF
FACSIMILE CODING TECHNIQUES SUBMITTED

TO THE CCITT BY GREAT BRITAIN AND GERMANY
October, 1979

FINAL REPORT

Submitted to:

NATIONAL COMMUNICATIONS SYSTEMS 8th & S. COURTHOUSE RD. ARLINGTON, VIRGINIA 2204

CONTRACTING AGENCY:

DEFENSE COMMUNICATIONS AGENCY

Purchase Order: DCA 100-79-M-0209

Submitted by:

DELTA INFORMATION SYSTEMS, DIC. 259 WYNCOTE ROAD JENKINTOWN, PENNA. 19046

TABLE OF CONTENTS

1.0	Introduct	tion
è.0	Measureme	ent Parameters
	2.1 1	Test Documents
	2.2	Resolution
	2.3	Minimum Scan Line Time
	2.4	Fransmission Bit Rate
	2.5	Measurement of Compression
	2.6	Measurement of Error Sensitivity
3.0	Computer	Program Overview
	3.1	The Simulation Process
	3.2	Program Structure
4.0	Generalia	zed Error Detection and Correction Procedure 4-1
5.0	Assumption	ons related to Individual Algorithms 5-1
	5.1	British Post Office 5-1
	5.2	Federal Republic of Germany 5-1
6.0	Measureme	ent Results 6-1
7.0	Reference	es
		APPENDICES
Α.	British '	Post Office CCITT Contribution - No. 77
в.		Republic of Germany CCITT Contribution - No. 82
		nes which are Common to all Algorithms
D.		t - British Post Office
E.		ing - British Post Office
		t - Federal Republic of Germany
		ing Endowel Bonublic of Company

1.0 INTRODUCTION

Several organizations have submitted contributions to the CCITT (see Appendices A, B, and References 4, 5, 9, 10, 11) describing two-dimensional coding techniques for selection of a standard compression algorithm for advanced digital facsimile systems. At the December 1978 meeting in Geneva, a working party of CCITT Study Group XIV adopted specific procedures to measure compression and error sensitivity so that candidate coding techniques may be compared on a meaningful basis. These definitions and procedures are outlined in references 1 and 2. The National Communications System of the U.S. Government has issued three contracts to Delta Information Systems, Inc. to evaluate seven candidate two-dimensional coding techniques using the criteria recommended by the CCITT.

In the first contract (Purchase Order DCA-79-M-0105), a basic computer program was developed to measure the compression and error sensitivity of digital facsimile coding techniques. To validate this program, the Modified-Huffman code, recommended as the one-dimensional standard for Group 3 machines, was tested and simulated on the model. The computer program and work accomplished on this initial contract is described in a Final Report issued August 10, 1979 (see Reference 3).

In the second contract, the validated computer model was used to measure the compression and error sensitivity of five two-dimensional coding techniques. The five coding algorithms which were selected for simulation were proposed by Japan, 3M, IBM, XEROX, and AT&T. These coding techniques were selected simply because no other contributions had been submitted to the CCITT when this NCS measurement contract was initiated. Contributions were subsequently submitted to the CCITT by the Federal

Republic of Germany and the British Post Office. The NCS organization issued a third contract (Purchase Order DCA 100-79-M-0209) to Delta Information Systems to measure the compression and error sensitivity of these latter two coding techniques and the results of this investigation are included in this document.

The measurement parameters which were involved in this program are summarized in Section 2.0 of this report. Section 3.0 describes the hierarchy and interrelationship of computer programs which are used in the measurement process. In many instances, the proposed operation of the coding algorithm was not totally defined when a transmission error was encountered. Section 4.0 describes the generalized error detection and correction procedure which was employed. As the computer programs were prepared for each algorithm, certain assumptions were made for each coding technique, particularly in the area of error detection and correction. These assumptions made for each individual coding technique are documented in Section 5.0.

Five separate computer runs were implemented for each algorithm at different combinations of test document, vertical resolution and K-factor. Section 6.0 summarizes the results of these measurements in terms of compression data, error sensitivity data, and coded line length statistics. Section 7.0 contains a list of reference documents related to the contract.

The CCITT contributions describing the two coding algorithms have been included in Appendices A and B for reference purposes. Appendix C contains the program code listings for those subroutines which are common to all algorithms, e.g. data packing, data unpacking, error measurement, etc. Appendices D, E, F, and G contain the flow charts and the listing

of the code for the computer program for the two algorithms which were investigated.

Delta Information Systems wishes to acknowledge the Contracting Officer's Technical Representative, Dennis Bodson, for the extraordinary level of support he has provided during the course of this contract. The assistance of Marla Thomas, from the DCEC computer facility, is also greatly appreciated.

2.0 MEASUREMENT PARAMETERS

In this section, the various parameters involved in the measurement of compression and error sensitivity will be summarized. In general, Study Group XIV of the CCITT agreed upon these measurement parameters at the general meeting held in Geneva in December 1978 (see Reference 2).

2.1 Test Documents

The test documents were chosen from the eight CCITT test documents (see Figure 2-1) since they have been widely used by data compression experimenters in the past. Documents numbered 1, 4, 5, and 7 (see Figures 2-2, 2-3, 2-4, and 2-5 respectively) were selected as the standard test images since these were considered most representative of documents to be transmitted.

The French PTT Administration has scanned the eight CCITT documents at the high resolution specified for Group 3 machines--7.7 lines/rm. They have also quantized each pel to be either black or white and stored the resultant image on magnetic tape. This tape was used as the source of input documents in this simulation program. Appendix B of Reference 3 describes the format of the test document magnetic tape supplied by the French PTT.

2.2 Resolution

It was agreed that measurements would be performed at both standard resolution (3.85 lines/mm.) and high resolution (7.7 lines/mm.). In the high resolution case, all lines on the input test documents shall be used. In standard resolution tests, every odd scan line

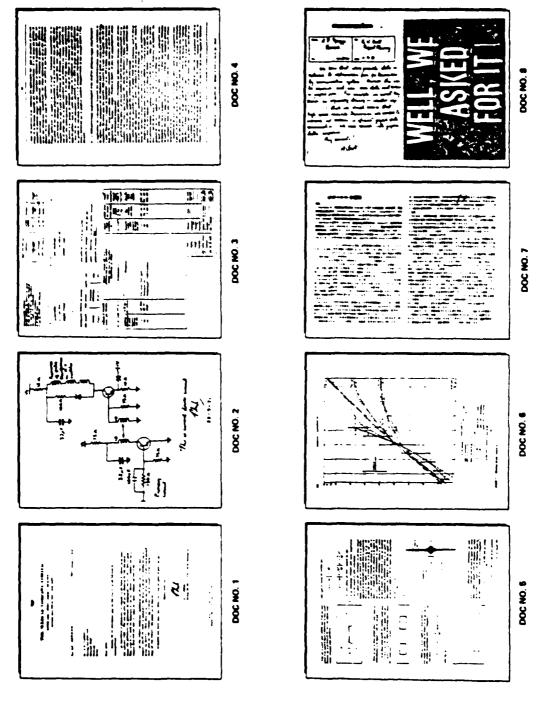


Figure 2-1 CCITT Standard Test Documents

SAPORS LANE - BOOLE - DORSET - BH 25 8 ER TELEPHONE BOOLE (945 13) 51617 - TELEX 123456

Our Ref. 350/PJC/EAC

18th January, 1972.

Dr. P.N. Cundall, Mining Surveys Ltd., Holroyd Road, Reading, Berks.

Dear Pete,

Permit me to introduce you to the facility of facsimile transmission.

In facsimile a photocell is caused to perform a raster scan over the subject copy. The variations of print density on the document cause the photocell to generate an analogous electrical video signal. This signal is used to modulate a carrier, which is transmitted to a remote destination over a radio or cable communications link.

At the remote terminal, demodulation reconstructs the video signal, which is used to modulate the density of print produced by a printing device. This device is scanning in a raster scan synchronised with that at the transmitting terminal. As a result, a facsimile copy of the subject document is produced.

Probably you have uses for this facility in your organisation.

Yours sincerely,

P.J. CROSS

Group Leader - Facsimile Research

Figure 2-2 CCITT Test Document No. 1

Registered in England: No. 2038
Registered Office: 80 Vicars Lane, liford, Essex.

L'ordre de lancement et de réalisation des applications fait l'objet de décisions au plus haut niveau de la Direction Générale des Télécommunications. Il n'est certes pas question de construire ce système intégré "en bloc" mais bien au contraire de procéder par étapes, par paliers successifs. Certaines applications, dont la rentabilité ne pourra être assurée, ns seront pas entreprises. Actuellement, sur trente applications qui ont pu être globalement définies, six en sont au stade de l'exploitation, six autres se sont vu donner la priorité pour leur réalisation.

Chaque application est confiée à un "chef de projet", responsable successivement de sa conception, de son analyse-programmation et de sa mise en oeuvre dans une région-pilote. La généralisation ultérieure de l'application réalisée dans cette région-pilote dépend des résultats obtenus et fait l'objet d'une décision de la Direction Générale. Néanmoins, le chef de projet doit dès le départ considérer que son activité a une vocation nationale donc refuser tout particularisme régional. Il est aidé d'une équipe d'analystes-programmeurs et entouré d'un "groupe de conception" chargé de rédiger le document de "définition des objectifs globaux" puis le "cahier des charges" de l'application, qui sont adressés pour avis à tous les services utilisateurs potentiels et aux chefs de projet des autres applications. Le groupe de conception comprend 6 à 10 personnes représentant les services les plus divers concernés par le projet, et comporte obligatoirement un bon analyste attaché à l'application.

II - L'IMPLANTATION GEOGRAPHIQUE D'UN RESEAU INFORMATIQUE PERFORMANT

L'organisation de l'entreprise française des télécommunications repose sur l'existence de 20 régions. Des calculateurs ont été implantés dans le passé au moins dans toutes les plus importantes. On trouve ainsi des machines Bull Gamma 30 à Lyon et Marseille, des GE 425 à Lille, Bordeaux, Toulouse et Montpellier, un GE 437 à Massy, enfin quelques machines Bull 300 TI à programmes câblés étaient récemment ou sont encore en service dans les régions de Nancy, Nantes, Limoges, Poitiers et Rouen; ce parc est essentiellement utilisé pour la comptabilité téléphonique.

Al'avenir, si la plupart des fichiers nécessaires aux applications décrites plus haut peuvent être gérés en temps différé, un certain nombre d'entre eux devront nécessairement être accessibles, voire mis à jour en temps réel : parmi ces derniers le fichier commercial des abonnés, le fichier des renseignements, le fichier des circuits, le fichier technique des abonnés contiendront des quantités considérables d'informations.

Le volume total de caractères à gérer en phase finale sur un ordinateur ayant en charge quelques 500 000 abonnés a été estimé à un milliard de caractères au moins. Au moins le tiers des données seront concernées par des traitements en temps réel.

Aucun des calculateurs énumérés plus haut ne permettait d'envisager de tels traitements. L'intégration progressive de toutes les applications suppose la création d'un support commun pour toutes les informations, une véritable "Banque de données", répartie sur des moyens de traitement nationaux et régionaux, et qui devra rester alimentée, mise à jour en permanence, à partir de la base de l'entreprise, c'est-à-dire les chantiers, les magasins, les guichets des services d'abonnement, les services de personnel etc.

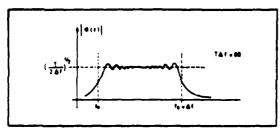
L'étude des différents fichiers à constituer a donc permis de définir les principales caractéristiques du réseau d'ordinateurs nouveaux à mettre en place pour aborder la réalisation du système informatif. L'obligation de faire appel à des ordinateurs de troisième génération, très puissants et dotés de volumineuses mémoires de masse, a conduit à en réduire substantiellement le nombre.

L'implantation de sept centres de calcul interrégionaux constituera un compromis entre : d'une part le désir de réduire le coût économique de l'ensemble, de faciliter la coordination des équipes d'informaticiens; et d'autre part le refus de créer des centres trop importants difficiles à gérer et à diriger, et posant des problèmes délicats de sécurité. Le regroupement des traitements relatifs à plusieurs régions sur chacun de ces sept centres permettra de leur donner une taille relativement homogèné. Chaque centre "gèrera" environ un million d'abonnés à la fin du Vième Plan.

La mise en place de ces centres a débuté au début de l'année 1971 : un ordinateur IRIS 50 de la Compagnie Internationale pour l'Informatique a été installé à Toulouse en février ; la même machine vient d'être mise en service au centre de calcul interrégional de Bordeaux.

Photo n° 1 - Document très dense lettre 1,5mm de haut Restitution photo n° 9

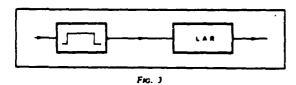
Cela est d'autant plus valable que $T\Delta f$ est plus grand. A cet égard la figure 2 représente la vraie courbe donnant $|\phi(f)|$ en fonction de f pour les valeurs numériques indiquées page précédente.



FKi. 2

Dans ce cas, le filtre adapté pourra être constitué, conformément à la figure 3, par la cascade :

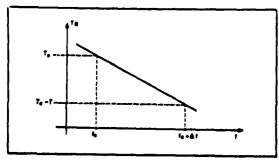
— d'un filtre passe-bande de transfert unité pour $f_0 \le f \le f_0 + \Delta f$ et de transfert quasi nul pour $f < f_0$ et $f > f_0 + \Delta f$, filtre ne modifiant pas la phase des composants le traversant;



— filtre suivi d'une ligne à retard (LAR) dispersive ayant un temps de propagation de groupe T_R décroissant linéairement avec la fréquence f suivant l'expression :

$$T_R = T_0 + (f_0 - f) \frac{T}{\Delta f} \quad (\text{avec } T_0 > T)$$

(voir fig. 4).



F16. 4

telle ligne à retard est donnée par :

$$\varphi = -2\pi \int_0^f T_R df$$

$$\varphi = -2\pi \left[T_0 + \frac{f_0 T}{\Delta f} \right] f + \pi \frac{T}{\Delta f} f^2$$

Et cette phase est bien l'opposé de $/\phi(f)$,

à un déphasage constant près (sans importance) et à un retard T_0 près (inévitable).

Un signal utile S(t) traversant un tel filtre adapté donne à la sortie (à un retard T_0 près et à un déphasage près de la porteuse) un signal dont la transformée de Fourier est réelle, constante entre f_0 et $f_0 + \Delta f$, et nulle de part et d'autre de f_0 et de $f_0 + \Delta f$, c'està-dire un signal de fréquence porteuse $f_0 + \Delta f/2$ et dont l'enveloppe a la forme indiquée à la figure 5, où l'on a représenté simultanément le signal S(t) et le signal $S_1(t)$ correspondant obtenu à la sortie du filtre adapté. On comprend le nom de récepteur à compression d'impulsion donné à ce genre de filtre adapté : la « largeur » (à 3 dB) du signal comprimé étant égale à $1/\Delta f$, le rapport de compression

est de
$$\frac{T}{1/\Delta f} = T\Delta f$$

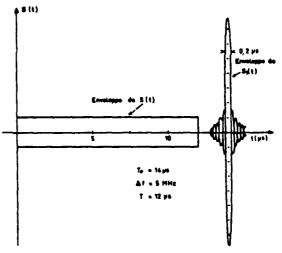


Fig. 5

On saisit physiquement le phénomène de compression en réalisant que lorsque le signal S(t) entre dans la ligne à retard (LAR) la fréquence qui entre la première à l'instant 0 est la fréquence basse f_0 , qui met un temps T_0 pour traverser. La fréquence f entre à l'instant $f = (f - f_0) \frac{T}{\Delta f}$ et elle met un temps

 $T_0 - (f - f_0) \frac{T}{\Delta f}$ pour traverser, ce qui la fait ressortir à l'instant T_n également. Ainsi donc, le signal S(t)

Figure 2-4 CCITT Test Document No. 5

CCITTの概要

沿革

の国際通信上の諸問題を真先に取上げ、その解決方法を見出して行く重要な機関での国際通信上の諸問題を真先に取上げ、その解決方法を見出して行く重要な機関で関連教皇録委員会、CCIR、CCITT)の一つとして、ITUの中でも、世界の国際通信上の指問題を真然の関係を表現を表現している。

CCITTの前身は、CCIF(国際電話諮問委員会)とCCIT (国際電話適問委員会」である。CCIFは、1925年の会観のとき、CCIFと併立するものとして設置され、これが1925年のパリ電信電話会観のとき、正式に、諮問委員会)が設置され、これが1925年のパリ電信電話会観のとき、正式に、間委員会)である。CCIFは、1924年にヨーロッパに「国際長距離電話適信問委員会)とCCIT (国際電話適信

ュネーブで、第4回総会は、1968年、アルゼンチンで開催された。し、第2回総会は、1960年にニューデリーで、第3回総会は、1964年、ジーは、同年同月に第8回総会が開催されたのち、併合されて現在のCCITTとなては、同年同月に第8回総会が開催されたのち、併合されて現在のCCITTとなるとして、CCIFは、1956年の12月に第18回総会が開催されたのち、CCI

である。 CCITTは、上述のように、ヨーロッパ内領域を想定したもの起する問題の研究が多い。 たとえば、1960年のCCITT動きの中で、技術上在でも、その影響を受け、会合参加国は、ヨーロッパの国が多く、ヨーロッパで生在でも、その影響を受け、会合参加国は、ヨーロッパの国が多く、ヨーロッパで生信・電話の技術・運用・料金の基準を定め、あるいは統一をはかってきたので、現信・電話の技術・運用・料金の基準を定め、あるいは統一をはかってきたので、現信・電話の技術・運用・料金の基準を定め、あるいは統一をはかってきたので、現代の国でにはよって、ヨーロッパ内の電

しい意見が導入されたことにも起因して、技術面、政治面の双方から導入されてき植民地の独立に伴ってITUの構成員の中にこれらの国が加わり、ITUの中に新至った。この汎世界的性格は第2次世界大戦後目ざましくなったアジア・アフリカを取り上げるに及び、CCITTの性格は漸次、汎世界的色彩を実質的に帯びるに電話通信の自動化および半自動化への技術的可能性を与え、CCITTがこの問題しかしながら、1956年9月に敷設された大西洋横断電話ケーブルは、大陸間

リー総会の単備文書で、この点には注目すべきであるとのべている。アメリカやアジアで総会が開催されたことがなく、CCITT委員長も、ニューデたことにもあらわれている。この総会までは、CCITでCCIFのいずれにしろ、た。CCITTの汎世界化は、1960年の第2回総会がニューデリーで開催された。CCITTの汎世界化は、1960年の第2回総会がニューデリーで開催され

任務

てみるならば、CCITTの任務は、つぎのとおりとなっている。れの機関の権限と任務は国際電気通信条約に明記されている。そこで条約を参照しITUは、全権委員会議、主管庁会議を始めとして、七つの機関をもち、それぞ

965年モントルー条約第187号) および料金の問題について研究し、および意見を表明することを任務とする。」(1および料金の問題について研究し、および意見を表明することを任務とする。」(1国所電信電話諮問委員会(CCITT)は、電信および電話に関する技術、運用

を払わなければならない。」(同第188号) 著に直接間違のある問題について研究し、および意見を作成するように妥当な注意にある国における地域的および国際的分野にわたる電気通信の創設、発達および改にある国际諮問委員会は、その任務の遂行に当たって、新しい国または発展の途上

について研究し、かつ、勧告を行なうことができる。」(同第189号) 「各国際諮問委員会は、また、関係国の要請に基づき、その国内電気通信の問題

上記集187号と第188号にいわれる「意見」とは、フランス語の Avis から上記集187号と第188号にいわれる「意見」とは、フランス語の Avis からまま世界の国際通信の活動方向であるともいえる。

は、関係国の意見を統一した国際的見解としては非常に便利である。ができ、また、その改正も容易であるので、現在のように進歩の早い国際通信界でって開催される主管庁会譲というような大会譲の決定をまたなくても表明することこの意見は、また、電信規則以下のその他の規則のごとく、数年以上の間隔をも

should be used. Figure 2-6 is a copy of the French PTT Test Document No. 4 scanned with 7.7 lines/mm. resolution. Figure 2-7 is a copy of the same document where the even scan lines have been replaced with the line above. Therefore, this represents a document in which the vertical resolution is 3.85 lines/mm.

2.3 Minimum Scan Line Time (MSLT)

The standard MSLT to be used in the measurement program will be 5, 10, and 20 ms. with EDL-code and 0 ms. without EDL-code. It was later clarified in a memo from the chairman of the Working Committee (see Reference 7) that if, for reasons of test economy, only one value of MSLT can be used in the test program, that value shall be 20 ms.

2.4 Transmission Bit Rate

The standard transmission bit rate is 4800 bits/sec.

2.5 Measurement of Compression

Two standard measures of compression have been established—

(1) number of coded bits (2) Compression Factor. The number of coded bits is the number of bits required to transmit a document, including all overhead bits such as End of Line (EOL) and Fill bits. The Compression Factor is computed by dividing the total number of picture elements (pels) per test document by the number of coded bits. It was further agreed that the Compression Factor and coded bits should be computed for two different conditions—with overhead and without overhead. The measurement with overhead applies to the

L'ordre de lancement et de réalisation des applications fait l'objet de décisions au plus haut niveau de la Direction Générale des Télécommunications. Il n'est certes pas question de construire ce système intégré "en bloc" mais bien au contraire de procéder par étapes, par paliers successifs. Certaines applications, dont la rentabilité ne pourra être assurée, re seront pas entreprises. Actuellement, sur trente applications qui ont pu être globalement définies, six en sont au stade de l'exploitation, six autres se sont vu donner la priorité pour leur réalisation.

Chaque application est confiée à un "chef de projet", responsable successivement de sa conception, de son analyse-programmation et de sa mise en oeuvre dans une région-pilote. La généralisation ultérieure de l'application réalisée dans cette région-pilote dépend des résultats obtenus et fait l'objet d'une décision de la Direction Générale. Néanmoins, le chef de projet doit dès le départ considérer que son activité a une vocation nationale donc refuser tout particularisme régional. Il est aidé d'une équipe d'analystes-programmeurs et entouré d'un "groupe de conception" chargé de rédiger le document de "définition des objectifs globaux" puis le "cahier des charges" de l'application, qui sont adressés pour avis à tous les services utilisateurs potentiels et aux chefs de projet des autres applications. Le groupe de conception comprend 6 à 10 personnes représentant les services les plus divers concernés par le projet, et comporte obligatoirement un bon analyste attaché à l'application.

II - L'IMPLANTATION GEOGRAPHIQUE D'UN RESEAU INFORMATIQUE PERFORMANT

L'organisation de l'entreprise française des télécommunications repose sur l'existence de 20 régions. Des calculateurs ont été implantés dans le passé au moins dans toutes les plus importantes. On trouve ainsi des machines Bull Gamma 30 à Lyon et Marseille, des GE 425 à Lille, Bordeaux, Toulouse et Montpellier, un GE 437 à Massy, enfin quelques machines Bull 300 TI à programmes câblés étaient récemment ou sont encore en service dans les régions de Nancy, Nantes, Limoges, Poitiers et Rouen ; ce parc est essentiellement utilisé pour la comptabilité téléphonique.

Al'avenir, si la plupart des fichiers nécessaires aux applications décrites plus haut peuvent être gérés en temps différé, un certain nombre d'entre eux devront nécessairement être accessibles, voire mis à jour en temps réel : parmi ces derniers le fichier commercial des abonnés, le fichier des renseignements, le fichier des circuits, le fichier technique des abonnés contiendront des quantités considérables d'informations.

Le volume total de caractères à gérer en phase finale sur un ordinateur ayant en charge quelques 500 000 abonnés a été estimé à un milliard de caractères au moins. Au moins le tiers des données seront concernées par des traitements en temps réel,

Aucun des calculateurs énumérés plus haut ne permettait d'envisager de tels traitements. L'intégration progressive de toutes les applications suppose la création d'un support commun pour toutes les informations, une véritable "Banque de données", répartie sur des moyens de traitement nationaux et régionaux, et qui devra rester alimentée, mise à jour en permanence, à partir de la base de l'entreprise, c'est-à-dire les chantiers, les magasins, les guichets des services d'abonnement, les services de personnel etc.

L'étude des différents fichiers à constituer a donc permis de définir les principales caractéristiques du réseau d'ordinateurs nouveaux à mettre en place pour aborder la réalisation du système informatif. L'obligation de faire appel à des ordinateurs de troisième génération, très puissants et dotés de volumineuses mémoires de masse, a conduit à en réduire substantiellement le nombre.

L'implantation de sept centres de calcul interrégionaux constituera un compromis entre : d'une part le désir de réduire le coût économique de l'ensemble, de faciliter la coordination des équipes d'informaticiens; et d'autre part le refus de créer des centres trop importants difficiles à gérer et à diriger, et posant des problèmes délicats de sécurité. Le regroupement des traitements relatifs à plusieurs régions sur chacun de ces sept centres permettra de leur donner une taille relativement homogène. Chaque centre "gèrera" environ un million d'abonnés à la fin du Vième Plan.

La miss en place de ces centres a débuté au début de l'année 1971 : un ordinateur IRIS 50 de la Compagnie Internationale pour l'Informatique a été installé à Toulouse en février ; la même machine vient d'être miss en service au centre de calcul interrégional de Bordeaux,

L'ordre de lancement et de réalisation des applications fait l'objet de décisions au plus haut niveau de la Direction Générale des Télécommunications. Il n'est certes pas question de construire ce système intégré "en bloc" mais bien au contraire de procéder par étapes, par paliers successifs. Certaines applications, dont la rentabilité ne pourra être assurée, ne seront pas entreprises. Actuellement, sur trente applications qui ont pu être globalement définies, six en sont au stade de l'exploitation, six autres se sont vu donner la priorité pour leur réalisation.

Chaque application est confiée à un "chef de projet", responsable successivement de sa conception, de son analyse-programmation et de sa mise en oeuvre dans une région-pilote. La généralisation ultérieure de l'application réalisée dans cette région-pilote dépend des résultats obtenus et fait l'objet d'une décision de la Direction Générale. Néanmoins, le chef de projet doit dès le départ considérer que son activité a une vocation nationale donc refuser tout particularisme régional. Il est aidé d'une équipe d'analystes-programmeurs et entouré d'un "groupe de conception" chargé de rédiger le document de "définition des objectifs globaux" puis le "cahier des charges" de l'application, qui sont adressés pour avis à tous les services utilisateurs potentiels et aux chefs de projet des autres applications. Le groupe de conception comprend 6 à 10 personnes représentant les services les plus divers concernés par le projet, et comporte obligatoirement un bon analyste attaché à l'application.

II - L'IMPLANTATION GEOGRAPHIQUE D'UN RESEAU INFORMATIQUE PERFORMANT

L'organisation de l'entreprise française des télécommunications repose sur l'existence de 20 régions. Des calculateurs ont été implantés dans le passé au moins dans toutes les plus importantes. On trouve ainsi des machines Bull Gamma 30 à Lyon et Marseille, des GE 425 à Lille, Bordeaux, Toulouse et Montpellier, un GE 437 à Massy, enfin quelques machines Bull 300 TI à programmes câblés étaient récemment ou sont encore en service dans les régions de Nancy, Nantes, Limoges, Poitiers et Rouen ; ce parc est essentiellement utilisé pour la comptabilité téléphonique,

A l'avenir, si la plupart des fichiers nécessaires aux applications décrites plus haut peuvent être gérés en temps différé, un certain nombre d'entre eux devront nécessairement être accessibles, voire mis à jour en temps réel : parmi ces derniers le fichier commercial des abonnés, le fichier des renseignements, le fichier des circuits, le fichier technique des abonnés contiendront des quantités considérables d'informations,

Le volume total de caractères à gérer en phase finale sur un ordinateur ayant en charge quelques 500 000 abonnés a été estimé à un milliard de caractères au moins. Au moins le tiers des données seront concernées par des traitements en temps réel.

Aucun des calculateurs énumérés plus haut ne permettait d'envisager de tels traitements. L'intégration progressive de toutes les applications suppose la création d'un support commun pour toutes les informations, une véritable "Banque de données", répartie sur des moyens de traitement nationaux et régionaux, et qui devra rester alimentée, mise à jour en permanence, à partir de la base de l'entreprise, c'est-à-dire les chantiers, les magasins, les guichets des services d'abonnement, les services de personnel etc.

L'étude des différents fichiers à constituer a donc permis de définir les principales caractéristiques du réseau d'ordinateurs nouveaux à mettre en place pour aborder la réalisation du système informatif. L'obligation de faire appel à des ordinateurs de troisième génération, très puissants et dotés de volumineuses mémoires de masse, a conduit à en réduire substantiellement le nombre.

L'implantation de sept centres de calcul interrégionaux constituera un compromis entre : d'une part le désir de réduire le coût économique de l'ensemble, de faciliter la coordination des équipes d'informaticiens; et d'autre part le refus de créer des centres trop importants difficiles à gérer et à diriger, et posant des problèmes délicats de sécurité. Le regroupement des traitements relatifs à plusieurs régions sur chacun de ces sept centres permettra de leur donner une taille relativement homogène. Chaque centre "gèrera" environ un million d'abonnés à la fin du Vième Plan.

La mise en place de ces centres a débuté au début de l'année 1971 : un ordinateur IRIS 50 de la Compagnie Internationale pour l'Informatique a été installé à Toulouse en février ; la même machine vient d'être mise en service au centre de calcul interrégional de Bordeaux. Group 3 situation while the measurement without overhead applies to the Group 4 case.

2.6 Measurement of Error Sensitivity

An objective measure of error sensitivity is obtained by encoding the test documents with the proposed techniques (all overhead bits must be included), subjecting the resulting bit stream to transmission errors, decoding the transmission to obtain the received image, and comparing the original image with the received image to determine the number of pels in error. The Error Sensitivity Factor (ESF) is calculated as the total number of document pels in error divided by the total number of transmission bits that are in error. In this way, the ESF represents the average disturbance to the output image caused by a single transmission error.

2.6.1 Transmission Error Pattern

It was agreed that a record of actual bit errors incurred over telephone lines will be used in the error sensitivity test. The Federal Republic of Germany (see Reference 8) has obtained a record of such errors by transmitting a known psuedo-random sequence at 4800 bits/sec. using a V27 ter modem over a switched telephone network. The resultant error pattern has been recorded on magnetic tape and made available to experimenters. Appendix C of Reference 3 describes the format of the transmission error magnetic tape. This tape was used in the measurement of error sensitivity described in this report.



2.6.2 Error Phases

One concern with the ESF measurement is the high degree of sensitivity to those few errors which may affect the end of line code and can cause an inordinate number of incorrect pels. If the error pattern happened to fall in an unfortunate phase relative to the encoded bits, a large number of pels could be affected. On the other hand, the error pattern could fall fortuitously and affect a relatively few number of pels. To insure experimenters can achieve an adequate level of statistical validity, the concept of error phases has been introduced. In the basic zero phase, the first bit of the error record is aligned with the first bit of the encoded transmission. In the case of Phase 2, the transmitted bit information is delayed by 1,024 bits relative to the previous run. The transmission bit information is delayed by 2,048 bits for Phase 2. Experimenters would have a higher confidence level in the average of the three phases compared to any one ESF taken alone.

2.6.3 Error Correction

In order to precisely measure the error sensitivity, both the encoding technique and the decoding algorithm must be completely defined. If more than one decoding algorithm is proposed (for example, to achieve differing levels of error control), each must be tested separately. Collective Letter No. 87 from the CCITT (see Reference 7) outlines an error correction procedure to be used for simulating two-dimensional algorithms where an error correction procedure has not been otherwise specified. In this procedure, the erroneous line is replaced

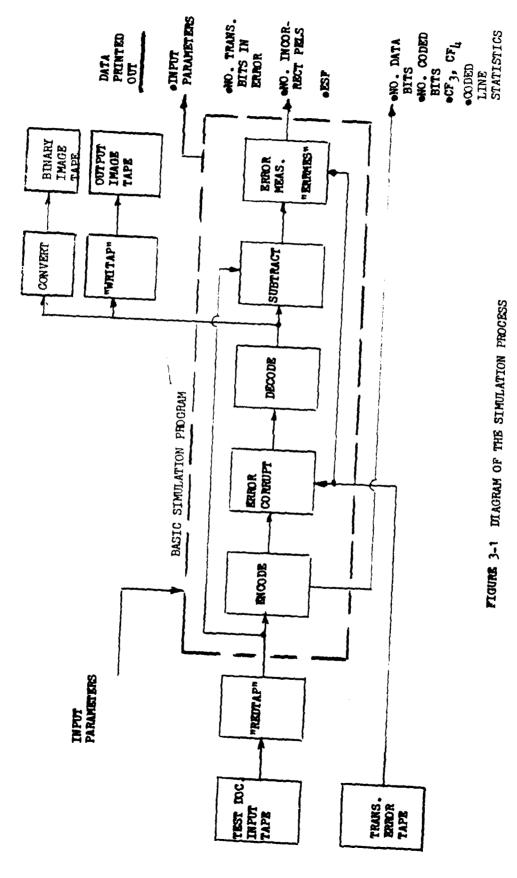
by the previous line and following lines are replaced by white lines until a one-dimensional coding line is correctly decoded.

3.0 COMPUTER PROGRAM OVERVIEW

This section contains a general overview of the computer program architecture used under this contract. The description is divided into two parts. Section 3.1 focuses on the overall simulation process from a flow perspective with particular emphasis on the simulation inputs and outputs. Section 3.2 presents the hierarchical structure of the programs illustrating how the programs are organized for each of the 7 different algorithms. For convenience of the reader, a detailed flow chart, and the actual program code listing, has been included in the Appendices for each algorithm (Appendices D through G). All computer programs have been written in conventional Fortran IV language.

3.1 The Simulation Process

Figure 3-1 illustrates the interrelationship between the major functions of each simulation program developed on the subject contract. There are two input data sets to each simulation which originate on magnetic tape. One tape, supplied by the French PTT Administration, contains all eight of the CCITT test documents. The format of this input image tape is described in Appendix B of Reference 3. The other tape, supplied by the Federal Republic of Germany, contains transmission error data from actual switched telephone circuits. The format of this input tape is described in Appendix C of Reference 3. A program called "REDTAP" was prepared to read the data from the input document tape while the error tape is read in directly. Data from the two input tapes are placed on disc in the computer system to be accessed during the simulation process. A separate file is established for each of the



test documents. The transmission error tape is divided into four files, one for each of four different circuit error conditions.

To initiate the simulation process, the operator must type in a set of input parameters. The insertion of the input parameters is accomplished on an interactive basis with prompting. A typical interactive sequence with responses is listed below.

- 1. PARAMETERS: INPUT (=I), OR DEFAULT (=D)? I
- 2. DIAGNOSTIC PRINTOUT? (Y OR N). N
- 3. ENTER MAXIMUM NUMBER OF PELS PER LINE: 1728
- 4. ENTER VERTICAL SAMPLING: 1
- 5. ENTER PARAMETER K: 4
- 6. ENTER ERROR PATTERN PHASE: 0
- 7. ENTER MINIMUM COMPRESSED LINE LENGTH: 96
- 8. NUMBER OF SCAN LINES TO BE PROCESSED = ? 10
- 9. ERROR MODE = ? (M=MANUAL, T=TAPE, N=NO ERRORS) N

After the data has been entered and the measurement parameters have been selected, the first step in the simulation process is the "ENCODE" function. This function detects color changes in the input data and constructs the appropriate code word by table look-up or algorithm. The actual code is fed to the error corrupt unit, while the number of code bits is accumulated with fill and EOL codes to provide the output total number of data bits, to compute the Compression Factors, CF_3 and CF_1 .

The error corruption step combines the transmission error data with the encoded data. At each point in the image where an error occurs, the corresponding bit in the encoded signal is reversed and fed to the

decode function. The decoder basically performs the inverse function of the encoder, generating a series of lines of image pels. There are two parts of the decoding function which are not obvious and require clarification: (1) what the decoder does when an error occurs (2) what the decoder does when a line is missing. The operation of the decoder under these two conditions is described in Section 4.

The output of the Decode function feeds the "WRITAP" or "CONVERT" functions for writing the error corrupted image on magnetic tape. It is also fed to a subtraction function which compares the decoded image with the original image. Pels which are in error are fed to the "ERRMES" subroutine which counts all the pels in the image which are in error. This subroutine also counts the number of transmission error bits which corrupted the encode signal. Finally, the "ERRMES" subroutine computes the ESF by dividing the number of incorrect pels by the number of transmitted bits in error.

Figure 3-1 shows that the simulation process provides a printout of all the computed performance data as well as a summary tabulation of the input parameters.

For more details on the computer programs, refer to Section 3.2 for a description of the program structure and to the Appendices for flow charts and program listings.

The reader should note that most of the software prepared under this contract is suitable for simulating any compression algorithm. The only subroutines which must be written specifically for a particular coding technique are the encode and decode subroutines.

3.2 Program Structure

The following section describes the structure of the computer program written to simulate the various algorithms. In addition, a brief description of each of the subroutines is given.

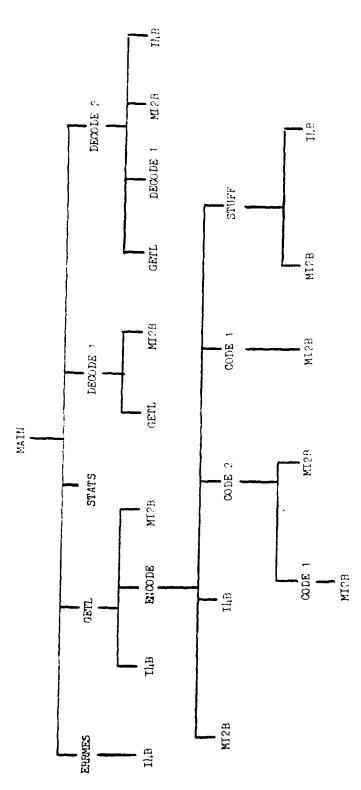
Each of the computer programs written to simulate the seven compression algorithms conforms to the general structure shown in Figure 3-2. The chart given in this figure shows the hierarchy of the functions that make up each simulation program. Some of the functions on the chart are named generically: the table in Figure 3-2 shows how these generic function names are keyed to the actual subroutine names used by each compression algorithm. The names on the hierarchical chart that do not appear in the table are subroutines that are used by all compression algorithms. A brief description of each of the functions/subroutines follows:

MAIN

The MAIN program controls the decoding process and the error recovery procedure for getting back in sync when an error is detected. As can be seen from Figure 3-2, the simulation process is "decode driven"; that is, the main program controls the decode process which decodes a buffered line of compressed data. When the contents of the buffer have been used up, a new line of data is encoded. The MAIN program also controls parameter input, measurement of errors, and reports computed results.

GETL

The GETL subroutine retrieves a number of requested bits from



THEFT		S	STREETTHE NAMES	ES			
POT TONG T	READ	УŒ	THE	XEROX	AT&T	1.53	Can
MALH	JPPEAD	THIMEN	JBM	XEROX	RTL	GEHMAT	HURIT LOSS
GETL	GETL	OETL 3	GETI.1	GETLX	GETLB	GETTA	# 144U
FI CODE	स्वकास	BICODS	EICODI	BICODX	ENCODE	ELCC Two	an one
CODE 1	CODELLI	CODELY	CODELIN	XCODLR	CODELM	COSTATE	
CODE 2	ООБЕУН	CODE3M	CODIBM	CODEX	CONRET	acopin	מוקהוט
DECODE 1	ONEDIM	ONED3	ONEIBM	OHEXOX	ONERT	57000	CODEGG
DECODE 2	TWDDIM	TWDD3	TWOIBM	TWDXOX	ntan Per	अवाजात	5:55 S

FIGURE 3-2 SUBROUTINE HIERARCHY

the coded line and delivers the bits packed into a word (right justified). If stuffing bits have been used, i.e. in the German code, they are removed. End-of-line codes (EDL) or line synchronization signals (LSS) are detected. If the number of coded bits requested by the calling program is not available, the ENCODE subroutine is called to provide them.

ENCODE

This subroutine supplies a line of compressed data. Color transitions on an input line are detected bit-by-bit. Both one-dimensional and two-dimensional lines are encoded depending on the parameter K. The code word is generated by table look-up, or algorithm, as appropriate, and added to the coded line buffer via SODE 1 and/or CODE 2.

CODE 1

The subroutine CODE 1 is called by ENCODE to look up the Modified Huffman Code (MHC) corresponding to a given run length and color, and add the code word to the coded line buffer.

CODE 2

The subroutine CODE 2 performs a similar function for the two-dimensional case. Rased on a particular feature, the appropriate code word is generated by table look-up or algorithm and added to the coded line buffer. All code tables for both one-dimensional and two-dimensional codes are stored in labelled common which is initialized by a BLOCK DATA subprogram.

STUFF

The STUFF subroutine is used by the READ and German algorithms to insert 0's or 1's in the coded data stream in order to avoid ambiguities with the line synchronization signal. A '1' is inserted after every occurence of ten consecutive zeroes in the coded steam for the German algorithm.

DECODE 1

The DECODE 1 subroutine decodes the MHC. It extracts a set of n bits (n=3 initially) from the coded line and looks for a match with all code words of length n, increasing n until a match is found or the code table is exhausted. When and if a match is found, the indicated bits are constructed on the output line. Any errors detected in the decoding process, such as no match to code table, or line too long, are flagged.

DECODE 2

This subroutine performs the same function as DECODE 1 for the two-dimensional line.

MI2B and ILB

The subprograms MI2B and I4B are used to pack and unpack a set of bits into (or from) an array of words.

4.0 Error Detection/Correction Procedure

In Reference 7.0, the following error checking and processing procedure was specified by the CCITT for testing the proposed two-dimensional coding techniques:

- 1) Error checking If decoded signals are not exactly 1728 pels/line, the line is recognized as an erroneous line.
- 2) Error processing The erroneous line is replaced by the previous line and following lines are replaced by white lines until one-dimensional coding line is correctly decoded.

The error detection and correction procedures used in this simulation follow the spirit, if not the letter, of this directive.

Not all of the proposed algorithms produce a line pel count that can be checked against the correct 1728 pels per line. The error checking was expanded to include the detection of any condition that could not possibly occur in a correctly received transmission. Some examples of possible error conditions are:

- EOL occurs before 1728 pels have been written
- More than 1728 pels have been written before EOL is received
- No word in applicable code table matches received bit pattern
- Current line decoding references a run that does not exist in the previous line
- Pels are written to the left of the first pel on the line

Conditions that are only improbable, such as a line of pels that differs radically from the previous line, are not considered error conditions. Error conditions specific to each coding algorithm are discussed in Section 5.0.

The AT&T algorithm does not, strictly speaking, have a "one-dimensional coding line." Therefore, the error processing was extended, for this algorithm, to consider any line that can be decoded without an error condition as a correct line. In decoding lines that reference previous lines, the last correctly decoded line is used as the reference line, regardless of whether or not there are intervening error lines. It is believed that the chance of correctly decoding a line, following an error line that references a previous line, is extremely small.

Upon detection of an error condition, the decoder attempts to resynchronize by searching for the next unique Line Synchronization Signal (LSS). All but the AT&T algorithm have different codes for one-dimensional and two-dimensional lines. The state diagram for error recovery for these algorithms is shown in Figure 4-1. For the AT&T algorithm, the One-Dimensional Decode and the Two-Dimensional Decode states are identical, and detection of an EOL in the Search state causes a change to the Decode state, rather than staying in Search.

Following Reference 7, when an error condition is detected, the error line is replaced by the previous correct line, while successive error lines are replaced by all-white lines, until a line is decoded correctly. It should be pointed out that this procedure

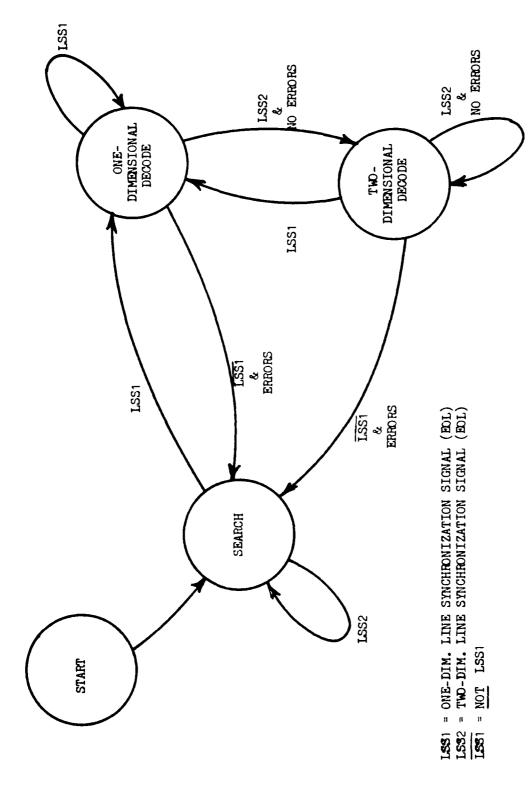


Figure 4-1 DECODE STATE DIAGRAM

may not be optimum. Repeating the last correct line until the next correct line is received may produce better results from a subjective and objective point of view.

Because of transmission errors, some of the original image lines may be missing in the output, or additional lines may be in the output that were not in the original image. In order that a missing or extra line not have an undue influence on the ESF, it is important that the original and received images not get permanently out of line alignment when they are compared to determine the number of pel errors. To this end, each of the lines in the original image is assigned a serial line number, and this number continues to be associated with the same line in the received image. If a transmitted line is dropped, due to the loss of an EOL, then its line number will be missing in the output. On the other hand, if a line is broken into two or more lines in the received image, due to false EOL's, then its line number will appear more than once in the output.

If no lines are dropped or added, the line numbers of the original and received lines that are compared to detect pel errors will be equal. When a line is added or deleted, the line numbers of the compared lines will become unequal. When this occurs for the first time, the two lines with different line numbers are compared to determine the number of pel errors, which is added to the pel error total. Then, instead of proceeding to the next line in both the original and received images, the next line is used in only one of the images, with the previous line being used in the other image. The line is advanced only in that image that has the smaller line number, so as to tend to make

the line numbers of the two images more equal. This continues until the line numbers are equal, after which the next line is used in both images, until another inequality is detected.

This procedure provides a proper penalty for a missing or added line, but prevents this type of error from causing pel errors over the entire image below the place where it occurred.

5.0 ASSUMPTIONS RELATED TO INDIVIDUAL ALGORITHMS

5.1 British Post Office

No modifications or assumptions were required to simulate the British algorithm. Two optional procedures, as defined in the British contribution, were simulated to enable best performance: The resettable K procedure was used to reset K when an all-white line followed a non-all-white line or a non-all-white line followed an all-white line. The optional step 3 in the two-dimensional coding procedure was used in place of step 2ii.

5.2 Federal Republic of Germany

The German compression algorithm encodes run lengths of correct predictions followed by an incorrect prediction. The runs between incorrect predictions are encoded separately for each source state. However, the contribution was not clear on the coding procedure if the last run of predictions on a line for a given source state did not end with a prediction error. Therefore, a hypothetical prediction error was added to each run of correct predictions for a given state if that run did not terminate naturally with a prediction error. This hypothetical error was automatically ignored on decoding. This procedure required a slight modification to the code table for state O (S_{\odot}). The code word 110 was used for a run length of 1729 instead of 1728 and the code word 10 for the prefix was used for lengths of 65-1728 instead of 65-1727.

Note that it is not possible with the German algorithm to detect errors by checking the decoded line length, since it is always 1728. Errors were detected by checking the residue of run lengths for each state after a complete line was decoded. For an error-free line, the residue must be 0 or 1 corresponding to runs ending with a prediction error and

runs ending with a hypothetical prediction error, respectively. Any runs of length greater than 1 "left over" after a line is decoded indicate an error. Taking the above approach, it should be noted that in the example of Figure 2 of Appendix B, the $S_{\rm O}$ run should be 6 instead of 5, since the first prediction error for state $S_{\rm O}$ occurs the sixth time state zero is present.

6.0 MEASUREMENT RESULTS

During the course of this contract, Delta Information Systems prepared computer programs to simulate the two-dimensional coding algorithms proposed by the British Post Office and the Federal Republic of Germany. These two programs were then run on the Hybrid Computer Facility at the Defense Communications Engineering Center in Reston, Virginia. Two different types of simulation were performed. The first measured the compression and error sensitivity of the two algorithms at five different test conditions (the four CCITT documents at standard resolution plus document number 4 at high resolution). In the second simulation, all seven proposed algorithms (Japan, 3M, IEM, Xerox, AT&T, BPO, FGR) were tested at an infinite K-factor for document number 4 at standard and high resolution. The results of these two simulation tests are summarized in the paragraphs below.

COMPRESSION AND ERROR SENSITIVITY

As explained above, five computer runs were performed for both the British and German algorithms. The following test conditions were held constant during these tests: error phase - 0; error file - 1; minimum scan line time - 20 ms. All four test documents (Documents 1, 4, 5, and 7) were run at standard resolution and a K-factor of 2. For the fifth run, test document number 4 was run at high resolution with a K-factor of 4.

The results of the ten test runs are tabulated in Tables 6-1 and 6-2. To aid in the evaluation process, the corresponding test data for the other five algorithms are also included in these tables. The definitions of measurement parameters included in these tables are reviewed below.

TABLE 6-1 COMPRESSION AND ERROR SENSITIVITY TEST RESULTS*

DOC. NO. VERT.RESOL. K FACTOR	ALGORITHM	NO.CODED BITS	NO.BITS IN EHROR XMTD	BER X 10-3	NO. INCORRECT PELS	NO. CODED DATA BITS	ESF	CF ₃	CF4
	JAPAN	442,434	362	.82	21,030	390,927	58.093	4.6399	5.2513
	3М	441,104	362	.82	12,255	397,549	33.8536	4.6539	5.1638
DOC.NO. 4	IBM	430,215	346	08.	16,013	383,562	46.2803	4.7717	5.3521
	XEROX	468,341	374	862.	15,642	430,660	41.8235	4.3833	4.7668
STANDARD	AT & T	466,613	374	.80	19,378	415,034	51.8128	4.3995	4.9463
RESOL.	вРо	442,129	362	.819	15,250	395,132	42.1271	4.6431	5.1954
K = 2	FRG	430,335	346	.804	15,674	385,149	45.3006	4.7704	5.3301
	JAPAN	727,418	564	.775	38,283	620,671	67.877	5.6442	6.6150
	3M	757,869	564	.74	38,682	668,555	68.5851	5.4175	6.1412
DOC.NO. 4	IBM	727,740	564	.77	30,600	627,122	54.2553	5.6418	6.5469
	хенох	822,790	564	689.	25,464	748,406	45.1489	4.9900	5.4860
нісп	AT & T	763,481	564	.73	33,756	655,807	59.8511	5.3776	6.2606
RESOL.	вро	731,769	564	.771	39,365	628,963	69.7961	5.6107	6.5278
K = 4	FRG	727,121	564	922.	33,293	631,072	59.0301	5.6466	6.5060
	JAPAN	188,070	120	.638	3,538	113,956	29.48	10.915	18.0145
	ЭМ	192,484	132	89.	1,160	126,122	8.7879	10.6651	16.2768
DOC.NO. 1	IBM	187,619	120	.63	3,034	115,011	25.2833	10.9417	17.8493
	XEROX	198,749	132	.664	2,571	133,050	19.4773	10.3289	15.4293
STANDARD	AT & T	193,573	132	89.	1,236	112,546	9.3636	10.6051	18.2402
RESOL.	ВРО	189,285	120	.634	3,091	115,540	25.7583	10.8454	17.7675
K = 2	FRG	189,938	120	.632	3,056	118,809	25.4667	10.8081	17.2787
				7	*				

MIN. SCAN LINE TIME - 20 ms. * FRROR PHASE - 0; ERROR FILE - 1;

TABLE 6-2 COMPRESSION AND ERROR SENSITIVITY TEST RESULTS (cont'd)

JAPAN 253,989 216 .850 7,549 210,040 34.94 7.7712 9.0 DOC.NO. 5 IBM 264,163 216 .81 7,386 226,815 34.1944 7.7712 9.0 DOC.NO. 5 IBM 264,163 216 .84 8,211 210,809 38.0139 8.0676 9.7 RESOL. XEHOX 269,544 220 .816 3,041 236,284 13.8227 7.6161 8.6 RESOL. BPO 265,470 220 .82 5,570 220,429 25.3182 7.6161 8.6 K = 2 FRG 255,470 216 .845 8,483 210,971 39.2731 8.0356 9.3 A = 2 FRG 258,815 216 .834 4,332 220,118 20.0556 7.9318 9.3 DOC.NO. 7 IBM 413,042 28 8,485 399,497 23.8343 4.7577 5.1 K = 2 IBMO 413,040	DOC.NO. & VERT.RESOL.	ALGORITIIM	NO.CODED BITS	NO.BITS IN ERROR XMTD	ВЕК Х 10-3	NO. INCORRECT PELS	NO. CODED DATA BITS	ESF	CF3	CF_4
AM 264,163 216 81 7,386 226,815 34.1944 7.7712 DC.NO. 5 IBM 254,459 216 .84 8,211 210,809 38.0139 8.0676 TANDARD XEROX 269,544 220 .816 3,041 236,284 13.8227 7.6161 RESOL. AT & T 267,503 220 .82 5,570 220,429 25.3182 7.6742 RESOL. BPO 255,470 216 .845 8,483 210,971 30.2731 8.0356 C. BO 258,815 216 .845 8,483 210,971 30.2731 8.0356 AT & T 423,040 290 .685 9,361 385,871 32.27 4.852 AT & T 413,042 272 .65 6,056 379,460 22.2647 4.9701 AT & T 45,171 362 .807 9,017 421,857 21,908 4.5740 AT & T 422,007 290 .687		JAPAN	253,989	216	.850	7,549	210,040	34.94	8.082	9.7737
$ \begin{array}{llllllllllllllllllllllllllllllllllll$		3M	264,163	216	.81	7,386	226,815	34.1944	7.7712	8.0508
FANDARD AT & T 269,544 220 .816 3,041 236,284 13.8227 7.6161 RESOL. AT & T 267,503 220 .82 5,570 220,429 25.3182 7.6742 C = 2 FRG 255,470 216 .845 8,483 210,971 39.2731 8.0356 A = 2 FRG 255,470 216 .845 8,483 210,971 39.2731 8.0356 A = 2 FRG 258,815 216 .845 9,361 385,871 32.27 4.852 3M 413,042 272 .65 6,056 379,460 22.2647 4.9701 ANDARU AT & T 448,809 362 .807 9,017 421,857 24,908 4.5501 SSOL. BIPO 422,007 290 .687 7,673 388,535 26.4586 4.8645 A = 2 FRG 290 .687 8,366 389,068 28.483 4.8645	DOC.NO. 5	IBM	254,459	216	. 84	8,211	210,809	38.0139	8.0676	9.7380
RESOL. AT & T 267,503 220 .82 5,570 220,429 25.3182 7.6742 C = 2 BPO 255,470 216 .845 8,483 210,971 39.2731 8.0356 C = 2 FRG 255,470 216 .834 4,332 220,118 20.0555 7.9318 JAPAN 423,040 290 .685 9,361 385,871 32.27 4.852 DC.NO. 7 1BM 413,042 272 .65 6,056 379,497 23.8343 4.7577 FANDARU AT & T 413,042 272 .65 6,056 379,460 22.2647 4.9701 SSOL. BPO 448,809 362 .807 9,017 421,857 24.9088 4.5740 SSOL. BPO 422,007 260 .687 7,673 388,535 26.4586 4.8645 FIG 422,096 290 .687 8,366 389,068 28.8453 4.8635	STANDARD	XEROX	269,544	220	.816	3,041	236,284	13.8227	7.6161	8.6881
K = 2 FRG 255,470 216 .845 8,483 210,971 39.2731 8.0356 JAPAN 423,040 290 .685 9,361 385,871 32.27 4.852 JAPAN 431,481 356 .82 8,485 399,497 23.8343 4.7577 JAPAN 413,042 272 .65 6,056 379,460 22.2647 4.9701 JAPAN 413,042 272 .65 6,056 379,460 22.2647 4.9701 JANDARU AT & T 451,171 362 .807 9,017 421,857 24.9088 4.5740 ESOL. BPO 422,007 290 .687 7,673 389,688 28.4848 4.8645 = 2 FRG 422,096 290 .687 8,366 389,068 28.84848 4.8645	RESOL.	AT & T	267,503	220	.82	029'9	220,429	25.3182	7.6742	9.3130
FEG 258,815 216 .834 4,332 220,118 20.0555 7.9318 JAPAN 423,040 290 .685 9,361 385,871 32.27 4.852 JAPAN 413,042 272 .65 6,056 379,460 22.2647 4.9701 XEROX 448,809 362 .807 9,017 421,857 24.9088 4.5740 FANDARU AT & T 451,171 362 .80 9,463 415,929 26.1409 4.8501 SSOL. BPO 422,097 290 .687 7,673 388,535 26.4586 4.8645 = 2 FRG 422,096 290 .687 8,366 389,068 28.8483 4.8635	,	ВРО	255,470	216	.845	8,483	210,971	39.2731	8.0356	9.7305
JAPAN 423,040 290 .685 9,361 385,871 32.27 4.852 3M 431,481 356 .82 8,485 399,497 23.8343 4.7577 DC.NO. 7 IBM 413,042 272 .65 6,056 379,460 22.2647 4.9701 XEMOX 448,809 362 .807 9,017 421,857 24.9088 4.5740 ESOL. BPO 422,007 290 .687 7,673 388,535 26.4586 4.8645 = 2 FIG 422,096 290 .687 8,366 389,068 28.8483 4.8635	Z = X	FRG	258,815	216	.834	4,332	220,118	20.0555	7.9318	9.3262
3M 431,481 356 .82 8,485 399,497 23.8343 4.7577 DC.NO. 7 IBM 413,042 272 .65 6,056 379,460 22.2647 4.9701 FANDARD AT & T 448,809 362 .807 9,017 421,857 24.9088 4.5740 ESOL. BPO 422,007 290 .687 7,673 388,535 26.4586 4.8645 = 2 FRG 422,096 290 .687 8,366 389,068 28.8483 4.8635		JAPAN	423,040	290	.685	9,361	385,871	32.27	4.852	5.320
DC.NO. 7 IBM 413,042 272 .65 6,056 379,460 22.2647 4.9701 TANDARD AT & T 448,809 362 .807 9,017 421,857 24.9088 4.5740 ESOL. BPO 422,007 290 .687 7,673 388,535 26.1409 4.8645 = 2 FRG 422,096 290 .687 8,366 389,068 28.8483 4.8635		38	431,481	356	.82	8,485	399,497	23.8343	4.7577	5.1386
FANDARD FANDARD FANDARD FANDARD FAT & T & 451,171 FINA FIRE FIRE FIRE FIRE FIRE FIRE FIRE FIRE	DOC.NO. 7	IBM	413,042	272	.65	6,056	379,460	22.2647	4.9701	5.4100
ESOL. BPO 422,007 290 .687 8,366 389,068 28.8483 4.8635		XEROX	448,809	362	.807	9,017	421,857	24.9088	4.5740	4.8663
= 2 FIG 422,096 290 .687 7,673 388,535 26.4586 4.8645 = 2 FIG 422,096 290 .687 8,366 389,068 28.8483 4.8635	STANDARD	AT & T	451,171	362	.80	9,463	415,929	26.1409	4.5501	4.9356
= 2 FIG 422,096 290 .687 8,366 389,068 28.8483 4.8635	KESOL.	ВРО	422,007	290	687	7,673	388,535	26.4586	4.8645	5.2836
	K = 2	FRG	422,096	290	.687	8,366	389,068	28.8483	4.8635	5.2764

- Coded Data Bits Total compressed bits required to transmit the document excluding all overhead bits - EOL, fill, etc.
- Coded Bits Total compressed bits required to transmit the document <u>including</u> all overhead such as EOL, fill, etc.
- CF₁ Number of document pels* divided by the number of coded data bits
- CF₃ Number of document pels divided by the number of coded bits
- BER Transmitted bits in error divided by the number of coded bits
- ESF Number of incorrect pels divided by the number of transmitted bits in error

CODED LINE LENGTH STATISTICS

The CCITT suggested that experimenters should measure the statistics related to the number of bits required to define the individual scan lines. Statistics which were measured are minimum bits/line, maximum bits/line, average bits/line, and standard deviation. Statistics were measured for each of the two algorithms and for each of the five test conditions. Table c-3 is a tabulation of the test results for a minimum scan line time of 20 ms.

DIFINITE K-FACTOR TEST RESULTS

The primary objective of this overall measurement program is to contribute to the selection of a standard two-dimensional coding technique for the Group 3 application. For this reason, attention has been focused

^{*}High Resolution - 2,376 lines X 1728 pels/line = 4,105,728 pels Standard Resolution - 1,183 lines X 1728 pels/line = 2,052,864 pels

TABLE 6-3 CODED LINE LENGTH STATISTICS*

		,				
	STANDARD DEVIATION	321.32	264.81	146.09	181.41	176.32
FRG	AVERAGE BITS/ LINE	362.10	305.88	159.80	217.76	355.18
H	MAXIMUM BITS/ LINE	1089	1089	797	1063	718
	MINIMUM BITS/ LINE	96	96	96	96	96
	STANDARD DEVIATION	335.72	272.77	149.69	183.33	178.37
ВРО	AVERAGE BITS/ LINE	372.10	307.95	159.27	214.98	355.16
BI	MAXIMUM BITS/ LINE	1231	1072	797	1045	718
	MININUM BITS/ LINE	96	96	96	96	96
VERTI- CAL	TION	3.85/pm	7.7/pm	3.85/pm	3.85/pm	3.85/pm
TEST DOCU-	MENT NO.	4	4	1	2	2

* MINIMUM SCAN LINE TIME - 20 ms.

on a low K-factor to permit satisfactory operation over noisy transmission channels. It is also anticipated that two-dimensional coding techniques will be employed in the future Group 4 situation where the communication error rate will be very low. In fact, the compression parameter CF_{4} was chosen to give some indication of performance in a Group 4 application. However, if the test results are to be truly representative for Group 4 operation, the K-factor should be increased. To provide data for this application, all seven candidate algorithms were tested for an infinite K-factor. Each algorithm was tested at both the standard and high resolution case. Table o-4 is a tabulation of the test results.

The reader will note that four of the coding techniques (Japan, IBM, Xerox, FRG) exhibit a very large error sonsitivity factor, while it is much lower for the others. All those algorithms exhibiting a large ESF cause the input image to turn all white when the first error occurs and it remains so to the bottom of the page. The other three techniques have some degree of automatic self correction for transmission errors. As a result the error sensitivity for these three algorithms is reduced.

NOMENCLATURE OF PRINTED ERROR-CONTAMINATED IMAGES

Independent of this contract, the National Communication System is printing the error-contaminated images which were simulated and listed in Tables 6-1, 6-2, and 6-4. Each of these printed images is labelled in accordance with a particular nomenclature. Table 6-5 is a list of the test parameters and corresponding image nomenclature for the FRG and BPO algorithms. This table is included to assist those readers who may wish to correlate the test results included herein with the NCS images.

TABLE 6-4 TEST RESULTS FOR INFINITE K-FACTOR*

VERT. RESOLUTION	ALGORITHM	NO.CODED BITS	NO.BITS IN ERROR XMTD	BER 3	NO. INCORRECT PELS	NO. CODED DATA BITS	ESF	$c\mathbf{F}_3$	CF4
	JAPAN	421,115	290	.589	249,247	363,284	859.47	4.8748	5.6509
	3.1	425,179	290	.682	16,652	381,510	57.42	4.8282	5.3809
STANDARD RESOL.	кат	399,045	220	.551	245,062	349,188	1,113.9	5.1444	5.8790
ა ა. ა.	хенох								
md./	AT & T	402,686	238	.591	31,493	350,103	132.3	5.0979	5.8636
	РРО	416,057	272	.654	35,666	365,761	131.1	4.9341	5.6126
	FRG	369,140	220	.551	245,092	352,379	1,114.05	5.1432	5.8257
	JAPAN	663,182	564	.850	504,457	550,527	894.4	6.1910	7.4578
	3.8	703,756	564	.801	698'96	613,946	171.75	5.8340	6.6874
HIGH RESOL.	Kal	664,554	564	.848	501,443	569,271	889.1	6.1782	7.3412
7.7	XEROX								
' ਮੁਰ <i>'</i>	L & LV	962, 396	564	.846	99,838	556,114	177.0	6.1620	7.3829
	Oda	661,948	564	.852	103,623	554,167	183.7	6.2025	7.4088
	FRG	663,011	564	.851	501,407	563,965	889.02	6.1925	7.2801

ERROR FILE - 1; MIN. SCAN LINE TIME - 20 ms. ERROR PHASE - 0; *DOCUMENT NO. - 4;

Table 6-5 Nomenclature of Printed Error Contaminated Images

Image * Nomenclature	CCITT Document Number	K-Factor	Vertical Resolution
188A	1	2	3.58
488а	Ţ	2	3.58
488B	Į.	4	7.7
588B	5	2	3.58
788 a	7	2	3.58
4881	L L	infinite	3.58
4882	Ц	infinite	7.7

^{*}The nomenclature has a BPO prefix for the British Post Office algorithm and a GRR prefix for the Federal Republic of Germany algorithm.

7.0 REFERENCES

- CCITT Contribution No. 66, "Criteria for the Evaluation of Two-Dimensional Coding Techniques for use in Digital Facsimile Terminals" Source: United States of America; Date: January 1979.
- 2. CCITT Contribution COM XIV No. 70, "Report of the Meeting Held in Geneva," 11-15 Dec. 1978, Annex No. 2, Section III.
- National Communications System Report, "Development of a Computer Program for Measuring the Compression and Error Sensitivity of Facsimile Coding Techniques," August 10, 1979.
- 4. CCITT Contribution COM XIV No. 42, Japan Algorithm.
- 5. CCITT Contribution COM XIV No. 74, 3M Algorithm.
- 6. National Communications System Report, "Measurement of Compression Factor and Error Sensitivity Factor of Five Selected Two-Dimensional Facsimile Coding Techniques," October 1979.
- 7. Collective Letter No. 87 from the CCITT to Members of Study Group XIV COM/TO dated 21 May 1979, page 5, section 4.0.
- 8. Federal Republic of Germany, "Sensibility of Redundancy Reducing Codes to Transmission Bit Errors," CCTTT Study Group XIV Contribution No. 5, February 1977.
- 9. CCITT Contribution COM XIV No. 64, IBM Algorithm.
- 10. CCITT Contribution COM XIV No. 84, XEROX Algorithm.
- 11. CCITT Contribution COM XIV No. 81, AT&T Algorithm.

APPENDIX A

CCITT STUDY GROUP XIV

Contribution No. 77

Source: British Post Office

International Telegraph and Telephone Consultative Committee (CCITT)

Period 1977-1980

COM XIV-No. 77-E

Original : English

Question: 2/XIV

Date: March 1979

STUDY GROUP XIV - CONTRIBUTION No. 77

SOURCE : BRITISH POST OFFICE

TITLE : PROPOSAL FOR OPTIONAL TWO-DIMENSIONAL CODING SCHEME FOR GROUP 3

FACSIMILE APPARATUS

.. <u>Introduction</u>

In Draft Recommendation T.4 (COM XIV, No 25, Annex 3, Dec 1977) which refers to Group 3 facsimile apparatus, paragraph 4.2 notes that the one-dimensional coding scheme may be extended as an option to a two-dimension coding scheme. This contribution proposes such a two-dimensional coding scheme called the R2 code, when is based upon the one-dimensional coding scheme given in Draft Recommendation T.-.

The R2 code uses a similar coding procedure to that of the READ code proposed by Japan (COM XIV, No 42, Nov 1978) but uses a different code table. Compared with the READ code, the R2 provides higher compression factors, is easier to implement and is expected to have a better performance in the presence of transmission errors.

2. Pesign of the R2 code

Best Available Copy

One of the most important factors concerning the choice of a 2-dimensional coding scheme is its sensibility to transmission errors. The one-dimensional coding scheme using a modified Huffman code includes a unique end-of-line (EOL) codeword '00000000001'. This codeword contains a number of redundant bits which ensures that this sequence of digits cannot occur naturally in the coded data stream. Therefore, an error occurring in a coded scan line cannot prevent the detection of the EOL codeword associated with that scan line. This restricts the damage caused by an error to a single line. Also, an error which corrupts one or more digits of the EOL codex itself may not necessarily prevent that EOL from being detected. This protection is achieved by decoding '0000000' as the end of a scan line. The subsequent coded weak line is then deemed to begin immediately following the next 'l' in the data atrons. For machines accommodating large paper widths and having upto 2560 picture elemen'... per line, the end of a scan line is recognized by detecting '00000000'.

- 2 -COM XIV-No. 77-E

The R2 two-dimensional coding scheme is designed to provide the same protection against the effects of errors. This is achieved by constructing the R2 code table so that it contains the codeword '0000000'. The remaining codewords are then added by considering the statistics of the various coding elements or modes. The complete table has the prefix property and is exhaustive (ie it is a Huffman code). Redundant bits are then added to the codeword '0000000' to form the required EOL codeword 11 x '0' + '1'. (A similar method was used in the design of the modified Huffman code tables specified for Group 3 machines). The R2 code table and corresponding code tree for the R2 code are shown in Table 1 and Figure 3 respectively.

There are a number of other differences between the R2 and READ codes. Computer simulation tests on the READ code (Section 4) indicate that the vertical coding elements $V_L(n)$ and $V_R(n)$, where n is greater than 3, occur infrequently compared with the other coding elements. Unlike the READ code, the R2 code uses horizontal mode coding in these cases. Hence the R2 code has a range of vertical mode coding of up to plus or minus 3 picture elements and the R2 code table contains specific codewords to represent the vertical coding elements $V_L(2)$, $V_R(2)$, $V_L(3)$ and $V_R(3)$. The R2 code does not include codewords equivalent to the READ code D(n) codewords.

The flow diagram for the R2 code is similar to that for the READ code except that an extra decision box (is $|a_1b_1| > 3$?) is inserted immediately before the decision box (is $[a_0a_1] > [a_1a_2]$?). The latter decision box is an adaptive coding procedure which ensures that certain changing elements on the coding line are coded by the most effecient means. The decision box is not essential in the R2 coding procedure and is increfore included as an optional procedure in the R2 flow diagram.

Table I shows that the EOL codeword is followed by a '1' or a '0' flag bit to indicate whether the next scan line is to be coded by one- or two-dimensional coding respectively. This allows the K parameter to be used in a flexible manner, called 'resettable K', as described in Annex 1, Paragraph 4.2.1b. (Note that the resettable K procedure can also be used with the READ code).

A formal description of the R2 coding procedure is given in the Annex in a format capable of being inserted in Paragraph 4.2 of Draft Recommendation T.4.

3 Comments concerning the R2 and read coding schemes

- There is no redundancy in the end-of-line codewords LSS1 and LSS2. Thus a transmission error which corrupts any of the digits LSS1 or LSS2 will prevent detection of that end-of-line codeword.
- The need to add a 'O' ("stuffing" bit) after the occurrence of five consecutive 'i's in the coded data stream obtained using the READ code increases the complexity of the coding and decoding processes. It also increases the transmission time for documents 1, 4, 5 and 7 by an average of 2.5%. On the other hand, stuffing bits are not required when the R2 code is used, since the end-of-line (EOL) codeword is unique.
- The R2 coding procedure is simpler than the READ coding procedure since the number of vertical coding elements is limited to seven. Hence, in the R2 code, codewords of the form D(n) are not required and the coder needs to consider only a small number of picture elements on the reference line when coding a changing element on the coding line.
- Step 3 is an optional step in the R2 code which does not affect the compatibility between machines. This step is an adaptive coding procedure represented by the decision box $\left[is\left[a_0a_1\right]>\left[a_1b_1\right]?\right]$ in the flow diagram. By omitting it, the R2 coding procedure is simplified since it is not necessary to code each changing element along the coding line by two different methods. The results show that the compression factors are not changed significantly if this decision box is omitted.

- 3 -COM XIV-No. 77-E

The existence of an error on a scan line of coded data transmitted using the 1-dimensional modified Huffman coding method can usually be detected since each decoded scan line between successive EOL codewords should consist of 1728 picture elements. This 1728 check can be used on every decoded scan line, whether or not "fill" bits have been transmitted. If an error is detected, then it is optionally possible for the receiver to apply some form of corrective action. For example the receiver may attempt to conceal the error by printing an all white line or the previous scan line.

However, the READ code does not always allow this 1728 check to be used to determine the occurrence of an error even when LSS1 or LSS2 has been correctly decoded. The problem is that "fill" bits can sometimes be erroneously decoded, for example, as a sequence of the codeword V(O), (see Table 3, Com XIV, No 42). In this case, the presence of an error may not be detected and error concealment could not be applied. The R2 coding algorithm avoids this problem by using a code table which will not allow "fill" bits to be decoded as valid data.

4 Results

The READ coding scheme, as described in ref 1, but with the addition of the resettable K procedure (see Annex, Section 4.2.1b), was simulated by computer program. This enabled the number of coded bits, both with and without bit stuffing and the frequencies of the various coding elements to be measured for four of the CCITT reference documents. Corresponding measurements were then obtained for the R2 coding scheme. Two sets of measurements were obtained, one which included the adaptive coding step 3 and one which omitted this step. Note that stuffing bits were not added to the coded data obtained using the R2 code.

the measurements were obtained for minimum line periods of 0, 5, 10 and 20 msees, which correspond to minimum numbers of bits per line of 0, 24, 48 and 90 bits respectively when transmission takes place at 4.8 kbits/sec. The resettable K procedure used to obtain these results was slightly different to that proposed as an option in the k. code. For these results, each all white scan line was one-dimensionally coded.

A useful comparison between the two codes can be obtained by considering the compression factors for the four documents measured with a minimum line period of 0 mases and including the appropriate LSS1/2 or EOL codewords. This shows that the addition of stuffing bits to the READ code increases the number of coded bits required by 2.5% on average (of Tables 2 and 3). Tables 4 and 5 indicate that the omission of the adaptive coding step 3 has very little effect upon the number of coded bits; the number of coded bits is slightly higher for documents 1 and 4 and slightly lower for the other two documents when step 3 is omitted. When compared to the READ code with bit stuffing, the R2 coding procedure requires, on average, 2.6% fewer coded bits (of Table 3 with Table 4 or 5).

The frequencies of the coding elements for the READ code are listed in Table κ . Table 7 shows some of the frequencies of the coding elements $V_R(n)$ and $V_L(n)$ for a equal to or greater than 2. This indicates that the number of elements where a ir greater than 4 is small relative to the frequencies of other coding elements. For comparison purposes, the frequencies of the coding elements obtained for the R2 code (including step 3) are shown in Table 8. It was found that the omination of step 3 had very little effect upon these statistics.

The results relate to documents recorded on a magnetic tape made available by the French Administration. Subsequent testing of the R2 and other codes will be perform according to the agreed test criteria using a new magnetic tape; which has been provided recently by the French Administration.

- 4 -COM XIV-No. 77-E

5. Conclusions

This contribution proposes that, by making a number of changes to the READ coding scheme, the performance of the scheme can be improved. These changes allow higher compression factors to be obtained, simplify the coding and decoding processes and may offer an improved performance in the presence of transmission errors. Further measurements are needed to determine the usefulness of the options, ie step 3 and resettable K, described in the R2 code:

- 5 -COM XIV-No. 77-E

ANNEX

THE R2 CODING SCHEME

4.2 Two-dimensional coding scheme

The two-dimensional coding scheme is an extension of the one-dimensional coding scheme specified in Paragraph 4.1.

4.2.1 One-dimensional coding

a. Fixed K Parameter

The first scan line is transmitted by one-dimensional coding. Also every Kth line following the first line is transmitted by one-dimensional coding to limit the vertical spread of damage caused by transmission errors. The following K-l lines are coded by two-dimensional coding.

The transmitter determines which lines are transmitted by one- or two-dimensional coding by adding a single flag bit after the EOL codeword as shown in Paragraph 4.2.2e.

b. Resettable K Parameter

This is an optional procedure which may be used to enable higher compression values to be obtained.

If one of the K-1 lines following the Kth line complies with either of the following conditions, then that line is transmitted by 1-dimensional coding and the value of K is again set equal to the K parameter.

- i. A scan line which is not all white but which follows an all white scan line.
- ii. An all white scan line which follows a scan line which is not all white.
- c. Value of the K Parameter

The value of the K parameter should be set as follows.

Normal resolution standard : K = 2Higher resolution standard : K = 4

d. One-dimensional coding method

This conforms with the description in Paragraph 4.1.

4.2.2 Two-Dimensional Coding

This is a line-by-line coding method in which the position of each changing picture element on the current or coding line is coded with respect to the position of a corresponding reference element situated on either the coding line or the reference line which lies immediately above the coding line. After the coding line has been coded it becomes the reference line for the next coding line.

a. Definition of changing picture elements

A changing element is defined as an element whose "colour" (ie black or white) is different from that of the previous element along the same scan line.

- The reference or starting changing element on the coding line. At the start of the coding line ao is set on an imaginary white changing element situated just before the first element on the line. During the coding of the coding line, the position of ao is defined by the previous coding mode (see Paragraph 4.2.2b).
- The next changing element to the right of ao on the coding line. This is the next element to be coded.
- a The next changing element to the right of a on the coding line.
- The first changing element on the reference line to the right of a and of opposite colour to a.
- b₂ The next changing element to the right of b₁ on the reference line.

b. Coding Modes

Une of three coding modes are chosen according to the coding procedure described in Paragraph 4.2.2c to code the position of each changing element along the coding line. Examples of the three coding modes are given in Figure 2.

i. Pass mode

This mode is identified when the position of b_2 lies to the left of a_1 . If the position of b_2 lies directly above a_1 , then this does not constitute a pass mode. When this mode has been coded, a_0 is set on the element of the coding line below b_2 in preparation for the next coding.

ii. Vertical mode

When this mode is identified, the position of all is coded relative to the position of bl. The relative distance a_1b_1 can take on one of seven values V(0), $V_R(1)$, $V_R(2)$, $V_R(3)$, $V_L(1)$, $V_L(2)$ and $V_L(3)$, each of which is represented by a separate codeword. The subscripts R and L indicate that all is to the right or left respectively of bl and the number in brackets indicates the value of the distance a_1b_1 . After vertical mode coding has occurred, the position of a_0 is set on al.

iii. Horizontal mode

When this mode is identified, both the runlengths a_1a_0 and a_1a_2 are coded using the codewords H + M(a_0a_1) + M(a_1a_2). H is the flag codeword 'Oll' taken from the 2-dimensional code table. M(a_0a_1) and M(a_1a_2) are codewords which represent the length and "colour" of the runs a_0a_1 and a_1a_2 respectively and are taken from the appropriate white or black modified Huffman code tables. After a horizontal mode coding, the position of a_0 is set on a_2 .

c. Coding Procedure

The coding procedure identifies the coding mode that is to be used to code each changing element along the coding line. An adaptive procedure may be used in some cases to determine which coding mode will provide the most efficient coding. When one of the three coding modes has been identified, an appropriate codeword is selected from the code table given in Table 1. The coding procedure is formally defined by the flow diagram given in Figure 1.

Step 1

i. If a pass mode is identified this is coded using the codeword '0001' (Table 1). Return to the start of the coding procedure.

ii. If a pass mode is not detected then proceed to Step 2.

Step 2

Determine the absolute value of the relative distance albl

1. If $|a_1b_1| > 3$ then transmit the distances a_0a_1 and a_1a_2 by horizontal mode coding (Paragraph 4.2.2b). Return to the start of the coding procedure.

ii. If $|a_1b_1| \le 3$ then transmit the relative distance a_1b_1 by vertical mode coding (Paragraph 4.2.2b). Return to the start of the coding procedure.

Step 3

This is an adaptive coding procedure which ensures that the most efficient coding mode is used to code the position of a_1 . This optional step replaces Step 2 ii).

If $|a_1b_1| \le 3$ then determine the value of $[a_1b_1]$, ie the number of bits required to code the relative distance a_1b_1 by vertical mode coding. Also, determine $[a_0a_1]$, the number of bits required to code the distance a_0a_1 by horizontal mode coding. This is equal to H + M(a_0a_1), where H is the flag codeword 'Oll' and M(a_0a_1) is the codeword taken from the appropriate modified Huffman code table and represents the "colour" and run-length value of a_0a_1 .

Case 1: $\begin{bmatrix} a_0 a_1 \end{bmatrix} > \begin{bmatrix} a_1 b_1 \end{bmatrix}$

Code a,b, by vertical mode coding. .

Code both the distances aoa and ala2 by horizontal mode coding.

The use of this optional step does not affect interworking between Group 3 facsimile machines.

- 8 -COM XIV-No. 77-E

- d. Coding the first and last picture elements on a line
 - i. The first run length on a line a_0a_1 is replaced by a_0a_1-1 . Therefore, if the first run is black and is deemed to be coded by norizontal mode coding, then the first codeword $M(a_0a_1)$ corresponds to a white run of zero length.
 - ii. The coding of the coding line continues until the position of the imaginary changing element situated just after the last actual element has been coded. This may be coded as a_1 or a_2 . Also, if b_1 and/or b_2 are not detected at any time during the coding of the line, they positioned on the imaginary changing element situated just after the last actual picture element on the reference line.
 - e. Line synchronization codeworu

To the end of every coded line is added the end-of-line (EOL) codeword '000000000001'. The EOL codeword is followed by a single flag bit which indicates whether one- or two-dimensional coding is used for the next line.

The flag bit is:-

- 1 : one-dimensional coding of next line
- O : two-dimensional coding of next line
- f. Fill bits

fill bits, consisting of variable length strings of '0's may be inserted before the EOL codeword as specified in Paragraph 4.1c.

g. Return to control

The format used is the same as specified in Paragraph 4.1d.

- 9 -COM XIV-No. 77-K

TABLE 1
The R2 code table

MONE	ELEMENT: COD		NOTATION	CODEWORD
PASS	b ₁ ,	p ^r	þ	0001
HORIZONTAL	* ₀ * ₁ ,	#1# ¹ *	н	011 + M(a ₀ a ₁) + M(a ₁ a ₂)
VERTICAL	w _l Just Under b _l	a ₁ b ₁ = 0	V(O)	1
	a _l on the	*161 - 1	V _R (a ₁ b ₁)	001
of.	of b ₁	n¹p¹ - 5		000011
		" ₁ b ₁ = 3		000001
	al on the	u ₁ b ₁ - 1	٧ _٢ (۵ ₁ 6 ₁)	010
	or pi	u ₁ b ₁ = 2		000010
		"1 ^b 1 = 3		0000001
ENU-	OF-LINE CODEWO	KN .	Eor	000000000001

A 'l' or a 'O' flag bit in added to the EOL codeword to indicate that the relies scan line is coded by one-dimensional coding or two-dimensional coding respectively.

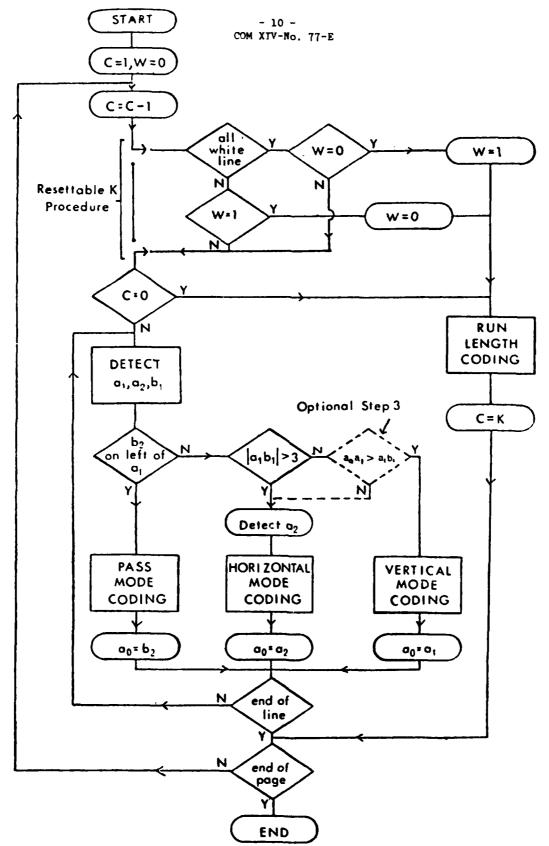


Figure 1 - Flow diagram for the two-dimentional coding scheme

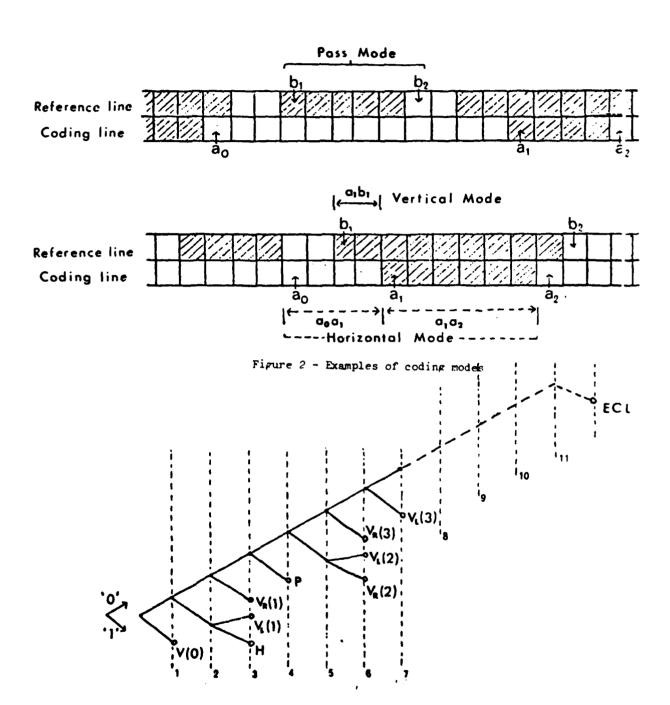


Figure 3 - The R2 code tree

- 12 -COM XIV-No. 77-E

(40%E: The results given in Tables 2 to 8 were obtained at a resolution of 3.85 lines per mm (ie .190 lines per document) and a resettable K parameter equal to 2.

TABLE 2

Number of coded bits - the read code without stuffing bits

DOCUMENT	O msecs WITHOUT LSS1/2	O msecs WITH LSS1/2	5 msecs	10 msecs	20 msec
1	111454	120974	.' 20992	138 656	18008a
.:	369379	378899	380661	390444	41444:
١,	190184	199704	200212	208225	231607
7	36183.	371352	371355	377361	<u> </u>

TABLE 3

Rumber of coded bits - the read code with stuffing bits

COCUMETT	O maecs WITHOUT LSS1/2	O msecs WITH LSS1/2	5 msecs	10 msecs	20 msec
,	114.40	123760	123778	141425	162763
4	380296	389816	391578	401349	425284
5	194043	.:035 63	204071	212068	235344
7	372251	381771	381774	387776	401837

TABLE 4

Number of coded bits - the R2 code (step 3 included)

DOCUMENT	O msecs WITHOUT EOL	O msecs WITH EOL	5 msecs	10 msecs	20 msecs
1	109952	125422	125430	139085	180042
4	361359	376829	377898	386194	409698
5	186752	202222	202384	208923	231495
7	354784	370254	370254	374906	388787

<u>Mumber of coded bits - the R2 code (step 3 omitted)</u>

DOCUMENT	O msecs WITHOUT EOL	O msecs W1TH EOL	5 msecs	10 msecs	20 msecs
l.	110173	125643	125651	139364	180364
4	362438	377908	378977	387292	41086 8
5	186309	201779	201941	208514	231150
7	353358	368828	368828	373484	387368

TABLE 6
Frequencies of coding elements - the read code

DOCUMENT	P	н	V(Q)	v _R (1)	V _L (1)	v _R (≥2)	V _L (≥ 2)
1	810	1814	4048	1315	1262	723	468
4	3640	7568	15217	5508	4577	2953	1911
5	1603	29 12	10240	2536	2457	972	734
7	4035	7470	13295	2801	5311	982	2335

Frequencies of the coding elements $V_R(n)$, $V_L(n)$,
where $n \ge 2$, for the read code

	<u>v</u>	here n > 2	, for the	read code		,
DOCUMENT	ν _R (∠)	۷ _L (2)	v _R (3)	۷ _L (3)	v _R (≥4)	V _L (≱ 4)
1	444	304	141	104	138	60
4	2016	1378	781	418	156	115
5	629	520	247	141	96	73
7	604	1518	236	421	143	396

TABLE 8

Frequencies of coding elements - the R2 code (including step 3)

DOCUMENT	p	н	V(0)	V _R (1)	V _L (1)	V _R (2)	ν _Γ (5)	v _R (3)	ν _L (3)
1	792	2130	3958	1279	1218	323	227	132	47
4	3565	8508	15008	5395	4265	1552	1042	6 96	128
5	1422	3266	10081	2591	2429	474	393	241	56
7	3858	8363	13141	2746	5134	481	946	228	263

APPENDIX B

CCITT STUDY GROUP XIV

Contribution No. 82

Source: Federal Republic of Germany

International Telegraph and Telephone Consultative Committee

(CCITT)

Period 1977-1980

COM XIV-No. 82-E

Original : English

Questions: 2/XIV - Point A.4

Date: March 1979

STUDY GROUP XIV - CONTRIBUTION No. 82

SOURCE : FEDERAL REPUBLIC OF GERMANY

TITLE : TWO-DIMENSIONAL CODING SCHEME

(Reply to Collective-letter No. 60)

Introduction:

A two-dimensional coding scheme for Group 3 facsimile apparatus is described as announced at the last meeting of Study Group XIV in Geneva, December 11. - 15., 1978. Differing from other proposals (IBM and Japan) this code makes use of a prediction method.

Annex 1 gives a detailed description of the code. Results of this code in comparison to other two-dimensional codes will be presented in a later contribution.

This code gives a very good performance, i. e. the compression factor is very high. Error susceptibility is comparable to other two-dimensicnal codes. This code uses a set of code word tables, which require a 256-word memory. One should take into account that implementation costs of memories are dropping constantly.

Besides high efficiency there is a distinct advantage of a clear patent situation. The owner of the respective patent will grant a duty-free licence to everybody. An obligatory declaration is given in Annex 2.

Annexes : 2

- 2 -COM XIV-No. 82-E

ANNEX 1

TWO-DIMENSIONAL CODING SCHEME

The two-dimensional coding scheme is a line-by-line coding method. It is an extension of the one-dimensional coding standard.

1. Parameter k

For reasons of restricting error propagation, one-dimensional coding with Modified-Huffman-Code is used for the first of every k lines. The parameter k can be chosen to k=2 for normal vertical resolution and k=4 for higher vertical resolution. Parameter k is set $k=\infty$, if transmission on data links with error control is used.

2. One-dimensional coding

One-dimensional coding of a scan line conforms with coding the run lengths by Modified-Huffman-Code.

3. Two-dimensional coding

The first step in the encoding process is to make a prediction of the present picture element X_0 from the neighbouring picture elements X_1 to X_4 (Fig. 1). Table 1 shows the predicted value X_0 depending on the four preceeding surrounding picture elements X_1 to X_4 . Each black-and-white pattern of these four picture elements defines a different source state S_1 . For each state there are individual conditional probabilities $P(X_0/S_1)$ that the present picture element X_0 will be white or black. Now the predicted value is the more probable one in the given state S_1 . Then the predicted value is compared to the real value of X_0 . Each time the prediction is right, a white pel is inserted for X_0 . When the prediction is wrong, X_0 is replaced by a

black pel. The resulting picture of prediction errors is a one-to-one transformation of the original picture, which means that all the information of the picture can be transmitted by coding the positions of the prediction errors.

The second step in the encoding algorithm explained here is to encode the run lengths between prediction errors as it is shown in Fig. 2. This is not done by coding the run lengths between every prediction error. The runs are encoded here separately for each source state S_j . For example the source is five times in state S_o until the first prediction error occurs in state S_o . So the run length to be transmitted is 5. For state S_1 there is a prediction error the third time the state S_1 occurs, so the run length is 3, etc.

For each state S_j an optimal run length code stored in an memory is used to transmit the run lengths between the prediction errors of state S_j (table 2). The storage of the run lengths codes requires a memory of 256 code words.

The run lengths codes used here are Truncated-Huffman-Codes, earlier described in /1/.

The encoder has to arrange the coded run lengths to be transmitted in a sequence corresponding to that of the states (Fig. 3). For example, first the encoder transmit the run length 5 of state S_0 since the current line starts with S_0 , then the run length 2 of state S_2 follows, afterwards the run lengths of state S_6 and state S_{15} etc.

Each scan line is terminated by a hypothetical prediction error at the end of the current line.

Each hypothetical picture element outside the page, which is requested for prediction, is assumed to be white. For example, for prediction of the first pel X_0 in an arbitrary line X_1 and X_4 are assumed to be white.

4. Line synchronisation

The line synchronisation signal used here conforms with the EOL-Code used by the Modified-Muffman-Code. A string of eleven "O" followed by a "1" is used. Additional one bit following the EOL-Code indicates one-dimensional or two-dimensional coding of the succeeding line. A "O" indicates one-dimensional coding, a "1" indicates two-dimensional coding. To make the line synchronisation signal unique, a "1" is inserted in the data stream after occurence of ten continuous "O"s.

5. Fill bits

Fill bits are used to obtain the minimum transmission time per line requested by the system. A variable string of "O"s is inserted in the EOL-Code.

6. Return to control

End of document transmission is indicated by six consecutive EOL-Codes.

7. Compression factor

Fig. 4 shows the compression factor CF_4 achieved by the two-dimensional coding with parameter $k=\infty$, no overhead for the eight testdocuments with 1728 pels/line and 2128 lines/page.

^{/1/} COM XIV, Doc. G3, No. 38
Dr.-Ing. Rudolf Hell GmbH
CCITT, Genf
October 1975

_	 	 			 ,	 •
		X ₄	x ₃	x ₂		previou
$\left\{ \right\}$		x ₁	x _o			current

s line line

 $X_1 =$ present picture element $X_1 - X_4 =$ previous picture elements

Figure 1 - Prediction pattern

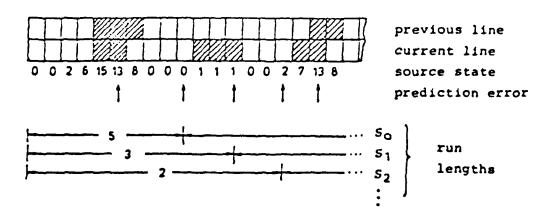


Figure 2 - Coding principle

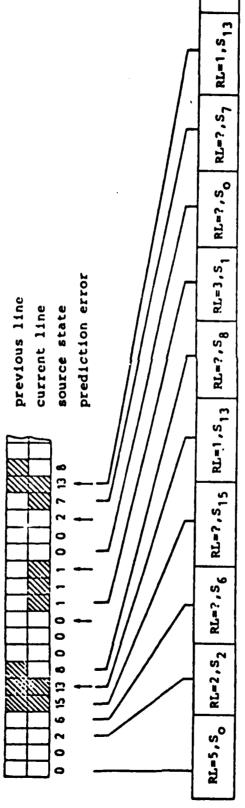


Figure 3 - Transmission sequence of run lengths (RL) between prediction errors

(2600)

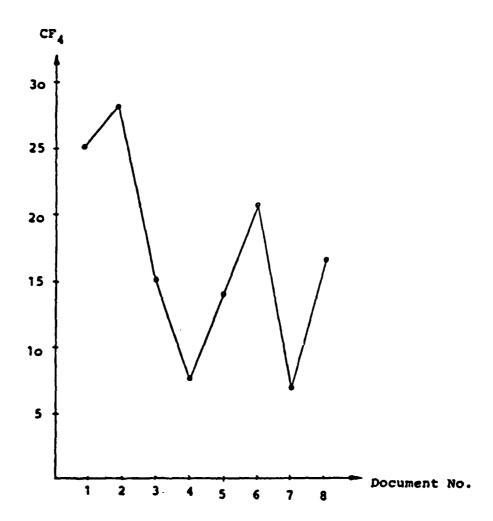


Figure 4 - Compression factor CF₄
1728 pels/line, 2128 lines/page
k = \omega , exclusive overhead

TABLE 1
Prediction Table

sta	ite S _j	predicted value X
s _o	X	white
sı		black
s ₂		white
s ₃		black
s ₄		White
s ₅		black
s ₆		black
s ₇		black
s ₈		white
s ₉		white
510		white
s ₁₁		black
512		white
s ₁₃		black
s ₁₄		white
s ₁₅	7284683 822	black

- 9 -COM XIV-No. 82-E

TABLE 2.1

Code Words

	urce utere S _o		source state 5 ₁₅
			13
length	endu word	run length	code word
1 1	OLLOLL	1 1	LOLL
1	o:co] 2	Loo
1	0000L 0101L	3	OLL LLO
3	111.	,	0001
1 6	LLLOO	1	OOLL
) i	LLLOL	1	LLLL
	ಂಬಂ	'8	00100
•	OLLLL	•	ororo
10	OOOLL	10	ororr
1 11	OOLLL	11	LLLOL
1 12	011000 000101	12	0000L0
113	000001	13	010001
i is	OOLOLL	1 13	LLLOOO
16	LLLLOOO	16	000000L
17	LLLLOOL	[17	001 0100
18	oroorro	10	00000L0
19	OLLOOLL	19	0000111
20	0100101 0100111	20 21	rotoror
22	OLLOLOL	22	FOLOTTO
23	0100010	23	OFOOFOF
24	orooroo	24	00000LLL
25	OLLOOLO	25	0000FF00
26	OFOOOFF	26	oororror
27	OLLLOLO	27	OOLOLOLL
28 29	00LL00L	28	00101111 00101111
30	0010100	30	0000000L
31	OOLLOLO	j , , , , , , , , , , , , , , , , , , ,	OLOOLLLO
32	OOLOLOL	32	OOOOLLOL
33	OOLLOOL] 33	OFOOFOOF
34	CL000000] 14	OLOOLLOO
35	01000001	35	OLOOLLLL
36	LLLLOLLO	36	reforme
34	LLLLOLOL	j j	OFOOFOOOF
39	OLLOLOOL	1 59	OOLOLLLOL
40	00010001	40	00000LL00
41	OLLLOLLL	41	TOTOTOOOO
42	0L111060L	42	LLLOOLLLL
43	00110110	43	LOLOLLLLL
44	000000LL	44	rinoilio
1 76	0000000L	46	OLOGLICIL
1 47	00110001	47	OOOOOLLOL
4.8	00LL0000	48	LOLDLLLLO
49	OOLLOLLL	49	oorororor
50	00000000	50	LOLOLOGOL
51	OLOGOOLOL	51 52	00000000L
52	011010000	52 53	0010111000
33	CLLCCOCL OLLCCLOCL	54	OOLOLLLOOL :
53	LLLLO1000	53	OLOOLLOLOO
56	OLLOLOCOL	56	00000000L
57	CLOCOCLID	57	oorororor
5.6	OTTTOPTOO	50 59	rotoropero
59	OLOOCOLLL	60	01001.00001
60	0L0000L00		OLOGILATION
1 62	0LLL00000	62	01'001'00'000
63	00000000L	63	0000000000L
64	000000000L	64	00000000000
1728	LLO	45-1728	LOLOO (prefix)
65-1727	LO (prefix)*		

- 10 -COM XIV-No. 82-E

TABLE 2.2

		Words	
800	rce etate S ₁ and S ₁₄	source	e state S ₂ and S ₁₃
zun lengen	code word	run length	code
,	ι	,	nLo
2	cro	3	LO
)	00L)	OOL
	011.0 000L	!	LLL
1	OLLLL	1	OLLL LLOO
,	or: roo	1 7	OLLOL
	000001	8	0000L
•	0000LL	9.	tiolo
10	0000101	10	00000L
1 12	00000011	11 12	000LL0
iii	00001000	1 13	000LLL
14	OLLLO1000	14	OLL0000
15	000000010	15	000000L
16	000010011	16	0001000 LLOLLIO
18	OLLLOLOOLL	1 18	OOOLOLL
19	000000L0to	19	OLLOGOLO
20	0000001311	20	OLLOOOLL
21	0000000111	21 22	0001010
13	0000100100	1 25	0000000L
24	0000001001	24	LLOLLLLO
25	00000000L	25	LLOLLLLL
26	0000100101	26	000100111
27	01110100101	27 28	000100110
29	01110100100	29	00000000L
30	00000010000	30	OOOLOLOLL
31	00000010001	31	000000000L
32-1728	Observation (prefix)	32-1728	000000000L LLOLLO (prefix)
•041	ce state S _g and S _g	* 0urce	s state \$3,\$4,\$5,\$7,\$8,\$10,\$11 and \$12
run length	code word	source run length	code word
£un .	code	Enu	code
run length 1	cole cole	run length 1	code word OLO L
run længth 1 2	Cole word L CL DOOL	gun length 1 2 3	code word OLO L OOL
run lungth 1 2 3	code word t ct coot coot	run length 1	code word OLO L
run længth 1 2	code word L CL SOUL GCLOS OCOCL	run length 1 2 3 4 5 6	code word OLO L OOL OOOL OLLLL OLLOOL
run lungth 1 2 3 4 5	code word L CL DOOL OOLL GCLOD OCOOCL DOOCL DOOCLD	run length 1 2 3 4 5 6	code word OLO L OOL OOL OLLL OLLOOL OCOOL OCOOL
run lwngth 1 2 3 4 5 6 7	code word t ct cooct	Fun length 1 2 3 4 5 6 7	code word OLO L OOL OOL OLLLL OLLOOL OOOOOL OOOOL
#un lwngth 1 2 3 4 5 6	code word L CL DOOL OOLL GCLOD OCOOCL DOOCL DOOCLD	run length 1 2 3 4 5 6	code word OLO L OOL OOL OLLL OLLOOL OCOOL OCOOL
xun lwngth 1 2 3 4 5 6 7 4 9	code vord L CL DOOL DOOL CCLO CCOOL CCCLO CCCCL DOOCL	Fun length 1 2 3 4 5 6 7 8 9	code word OLO L OOL OOL OLLLL OLLOOL OOOOL OOOOL OLLOOC OLLOOO OLLOOO OLLOOO
######################################	code word L CL DOOL DOOL GCLOD CCOOCL DOOCLO CCOCL DOOCLOL CCOCL DOOCLOL COCOCL DOOCLOL COCOCL COCOCC COCOC COCOCL COCOCC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COCOC COCOC COCOC COCOC COCOC COCOC COCOC COCOC COC COCOC COCOC COCOC COCOC COC C	Fun length 1 2 3 4 5 6 7 7 8 9 10	code word OLO L OOL OOL OOL OLLLL OLLOOL OOCOL OLLOOL OLLOOL OLLOOL OLLOOO OLLOOO OLLOOO OLLOOO OLLOOO
#un lwngth 1 2 3 4 5 6 7 8 9 10 11 12	code vord L CL CD CODE CODE CODE CODE CODE CODE CODE C	Fun length 1 2 3 4 5 6 7 8 9	code word OLO L OOL OOL OLLLL OLLOOL OOOOL OLLOOL OLLOOL OLLOOO OLLOOO OLLOOO OLLOOO OLLOOO OLLOOO OLLOOL OOOOOL
run lwngth 1 2 3 4 5 6 7 7 8 9 10 11 12 13 14 15	code word L CL DOOL DOOL GCLOD CCOOCL DOOCLO CCOCL DOOCLOL CCOCL DOOCLOL COCOCL DOOCLOL COCOCL COCOCC COCOC COCOCL COCOCC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COC COCOC COCOC COCOC COCOC COCOC COCOC COCOC COCOC COCOC COC COCOC COCOC COCOC COCOC COC C	Fun length 1 2 3 4 5 6 7 8 9 10 11 12 13 14	code word OLO L OOL OOL OLLLL OLLOOL OOOOL OLLOOL OLLOOC OLLOOO OLLOOO OLLOOO OLLOOOL OCLOOOL OCLOOOL OLLOOOL OLLOOOL OLLOOO OLLOOOL
######################################	code word L CL DOOL DOOL GCLOD GCOOL GCLOL GCLOC GCLC GCL	Fun length 1 2 3 4 5 6 7 7 8 9 10 11 12 13 14 15 16	code word OLO L OOL OOL OOL OLLLL OLLOOL OOCOL OLLOOL OLLOOC OLLOOL OLLOOC OLLOOL OLLOOC OLLOLLL OLLLOOC
#un lwngth 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 14	code word L CL DOOL DOOL GCLOO GCGCL DOOCLD GCGCLL DOCCOCL GCGCLL DOCCOCL GCGCLL DOCCOCL GCGCLL GCGCLC GCGCC GCCC GCCC GCCCC GCCC GCCC GCCC GCCC	run length 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17	code word OLO L OOL OOL OOLL OLLLL OLLOOL OOOOLO OLLOOOL OLLOOO OLLOOO OLLOOO OLLLOOL OLLOOO OLLLOOO OLLLOOO OLLLOOO OLLLOOO OLLLOOO OLLLOOO OLLLOOO OLLOLL
xun lwngth 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 14 15	code	Fun length 1 2 3 4 5 6 7 7 8 9 10 11 12 13 14 15 16	code word OLO L OOL OOL OOL OLLLL OLLOOL OOCOL OLLOOL OLLOOC OLLOOL OLLOOC OLLOOL OLLOOC OLLOLLL OLLLOOC
#un lwngth 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 14	code word L CL DOOL DOOL GCLOO GCGCL DOOCLD GCGCLL DOCCOCL GCGCLL DOCCOCL GCGCLL DOCCOCL GCGCLL GCGCLC GCGCC GCCC GCCC GCCCC GCCC GCCC GCCC GCCC	Fun length 1 2 3 4 5 6 7 7 8 9 10 11 12 13 14 15 16 17 18 19 20	code word OLO L OOL OOL OOL OLLLL OLLOOL OOCOL OLLOOL OLLOOC OLLOOC OLLOOC OLLOCO OCLLOCO OCLLOCO OCLLOCO OCOCLLO OCOCCLLO OCOCCLLO OCOCCLLO OCOCLLO OCO
run lwngth 1 2 3 4 5 6 6 7 8 9 10 11 12 13 14 15 14 15 16 17 18 19 20 21	code	run length 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21	code word OLO L COOL OOOL OCULL OLLOOL OCOCOO OLLOOO OLLOOO OLLOOO OLLOOO OLLOOO OLLOOO OLLOLO OCOCOOL OCOCOOL OCOCOOL OCOCOOL OCOCOOL OCOCOOL OCOCOOL OCOCOOL OCOCOOL OCOCOLLO OCOCOLLO OCOCLLL
xun lwngth 1 2 3 4 5 6 7 7 8 9 10 11 12 13 14 15 14 15 14 15 14 17 18 19 20 21	code	Fun length 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 14 15 16 17 18 19 20 21 22	code word OLO L COL COL COL COLL COLL COCOL COCOL COCOL COLLOCOL CLLCOC CCLLCOC CCLLCOC CCCLLCOC CCCLLCC CCCCCC CCCLLCC CCCCCC CCCCCC CCCCCC
run lwngth 1 2 3 4 5 6 7 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23	code	Fun length 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23	code word OLO L DOL OOL OOL OLLIL OLLOOL OOOOL OLLOOL OLLOOO OLLOOL OOOOOL OOOOLL OOOOOO
xun lwngth 1 2 3 4 5 6 7 7 8 9 10 11 12 13 14 15 14 15 14 15 14 17 18 19 20 21	code	Fun length 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 14 15 16 17 18 19 20 21 22	code word OLO L COL COL COL COLL COLL COCOL COCOL COCOL COLLOCOL CLLCOC CCLLCOC CCLLCOC CCCLLCOC CCCLLCC CCCCCC CCCLLCC CCCCCC CCCCCC CCCCCC
xun lwngth 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 14 15 14 15 20 21 22 23 24 25 26	code	Fun length 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 14 15 16 17 18 19 20 21 22 21 24 25 26	code word OLO L COL COL COL COLL COLL COLL COLL
run lwngth 1 2 3 4 5 6 7 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27	code	Fun length 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27	code word OLO L COOL OCLL COOL OCLLL OCLCOCL OCCCOCO OLLOCOL OLLOCOL OLLOCOL OLLOCOL OLLOCOL OLLOCOL OLLOCOL OLLOCOL OLLOCOL OCCCOCOL OCCCL OCCCCOCOL OCCCL OCCCCC OCCL OCCCCC OCCCL OCCCCC OCCCL OCCCCC OCCCCCC
run lwngth 1 2 3 4 5 6 7 7 8 9 10 11 11 12 13 14 15 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28	code	run length 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 21 24 25 26 27 28	code word OLO L COOL OOOL OOOL OLLUL OLLOOL OCOCOO OLLOOO OLLOOO OLLOOO OLLOOO OLLOLOO OLLOLO OCOCOOOOL OCOCOOOOL OCOCOOOOL OLLOLLO OCOCOOOOL OLLOLLO OLLOLLO OLLOLLO OLLOLLO OLLOLL
### ##################################	code	Fun length 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 24 25 26 27 28 29 30	code word OLO L COOL OCOL OCOL OCOL OCOL OCOCOC OCLUCOC OCLOCOC OCCLICOC OCOCOCOC OCOCILIO OCOCOCOC OCOCILIO OCOCOCOC OCLICICIO OCOCOCOC OCLICICIC OCOCOCOC OCLICICICIC OCOCOCOCOC OCLICICICIC OCOCOCOCOC OCOCOCOCOC OCLICICICICO OCOCOCOCOC OCOCOCOCOC OCLICICICOCOC OCOCOCOCOC OCLICICICOCOC OCOCOCOCOC OCLICICICOCOC
run lwngth 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 14 15 14 15 22 23 21 22 23 24 25 26 27 28 29 30	code word L CL DOOL DOOL DOOL DOOL DOOL DOOL DOO	run length 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 21 24 25 26 27 28 29 30 31	code word OLO L COOL COOL COLLIL OLLOOL COCOCO COLLOOO CLLOCO CLLOCO CLLOCO CLLOCO CLLOCO CLLOCO CLLOCO CLLOCO CLLOCO CLLOLL COCOCOCO
### ##################################	code	Fun length 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 24 25 26 27 28 29 30	code word OLO L COOL OCOL OCOL OCOL OCOL OCOCOC OCLUCOC OCLOCOC OCCLICOC OCOCOCOC OCOCILIO OCOCOCOC OCOCILIO OCOCOCOC OCLICICIO OCOCOCOC OCLICICIC OCOCOCOC OCLICICICIC OCOCOCOCOC OCLICICICIC OCOCOCOCOC OCOCOCOCOC OCLICICICICO OCOCOCOCOC OCOCOCOCOC OCLICICICOCOC OCOCOCOCOC OCLICICICOCOC OCOCOCOCOC OCLICICICOCOC

⁽²⁶⁰⁰⁾ The prefix is followed by the run length coded in 11-bit binary notation, most significant B-10 B-10

- 11 -COM XIV-No. 82-E

ANNEX 2

DR.-ING. RUDOLF HELL

GESELLSCHAFT MIT BESCHRANKTER MARTING

DR ING RUDOLF HELL GMBH POSTFACH 82 29 2300 HIEL 14

Fernmeldetechnisches Zentralamt der Deutschen Bundespost Referat A 26 Am Kavalleriesand 3 INFORMATIONSTECHNIK ELEKTRONIK FÜR SATZ UND REPRODUKTION

6100 Darmstadt

IHR JEICHEN UND IHRE NACHRICHT WI'M

UNSER ZEICHEN

DURCHWAHL

GRENZSTR 15 2100 HIEL 14

Lf/Hbs.

309

19th March, 1979

The following is a translation of a declaration of our company which was directed to the FTZ on November 10, 1978.

A copy of the original German text is attached.

Declaration

In the case that the method of a two-dimensional coding according to our German patent no. 25 56 803 "Process for Data Compression of Binary Coded Picture Signals" should be part of a CCITT recommendation, we irrevocably commit ourselves to grant by request a duty free licence to the above mentioned patent to everybody. This declaration will also be valid for our legal successors.

DR.-ING. RUDOLF HELL GMBH signed by Taudt Marhencke

WERR DIETRICHSDORF TEL 104311 2001 1 TELET C297858 FAT 108 111 2001 447
WERR SUCHSDORF TEL 104311 2013 1 FAT 104311 3013714 TELET C297858 FAT 108 111 2001 447
VORSITZENDER DES AUSSICHTSRATES DR ING DR ING E H RUDOLF HELL GESCHAFTSFUHRER DR RER NAT ROLAND FLOSS
DIPL REM ERNST FRICH MARMENCRE DIPL ING MEINZ TAUDT SITZ DER GESELLSCHAFT RIEL HR AMTSGERICHT RIEL ART B NE 61.

(2600)

APPENDIX C

SUBROUTINES WHICH ARE

COMMON TO ALL ALGORITHMS

APPENDIX C

SUBROUTINES WHICH ARE COMMON TO ALL ALGORITHMS

PROGRAM NAME	FUNCTION	PAGE
REDTAP 32	.Read input image tape	C-1
CODELN	Line Code Subroutine of "Encode" Subroutine	C-2
STATS	.Computes Statistics of Coded Lines	.c-3
BLOCK DATA	Initializes Packing/Unpacking Masks	C-4
MI2B	Packing Subroutine	.C-5
14в	.Unpacking Subroutine	.c -6
ERRMES	Error Measurement Subroutine	C-7
WRITAP 32	Converts binary data to Input Format	C-9
CONVERT	.Converts binary data to IBM Printer Format	C-10

```
START OF DOES UPRINT PROGRAM DSNAME=D0031.REDTAP.FORT
                     PROGRAM REDTAP32
                      IMPLICAT INTEGER(4-Z)
INTEGER PELBUF(1500).OTBUF(60)
DATA PELMAX.PELFIL.OTFIL.TERM/1728.1.2.5/
Č
                      INLNCT=0
       150 CONTINUE
       DO 100 I=1.60
                       19=1
                       IF=250
       READ(P9.FIL.300.END=500) IC.J
300 FOR4AT(25014)
       J1=J
316 IF(J.GT.250) GO TO 315
                       JIJI=J+IJ-1
READ(PELFIL,330) (PELBUF(K),K=ID,JID1)
                      GO TO 400
                                                                      and the second of the second o
       315 CONTINUE
                       READ(PELFIL, 300) (PELBUF(K), K=ID, IF)
                       ID= IF+1
IF=IF+250
                        J= J- 250
       IF(J.EQ.0) GO TO 400
GG TO 316
400 CONTINUE
       IF (INLNCT.GT.200) GO TO 450
#RITE(TERM.410) IC.J1
410 FURMAT(5X.I4.5X.I6)
#RITE(TERM.420) (PELBUF(K).K=1.J1)
C
        420 FORMAT (2X, 20(14, 2X))
       450 CONTINUE
                      OTELP=1
                      00 460 I=1.J1
RUN=PELBUF(1)
                       IF (RUN .EQ .O ) GC TO 700
      OF TRUN-EGOUT GE TO 700
DO 470 K=1.RUN
CALL MI2B(IC.OTBUF.GTELP.1)
OT ELP=OTELP.1
IF(JTELP.GT.PELMAX) GO TO 480
470 CONTINUE
IC=MOD(IC+1.2)
460 CONTINUE
        480 CUNTINUE
                        INLNCT=INLNCT+1
                       #RITE(OTFIL) INLNCT, PELMAX, DTBUF
GO TO 150
        500 CONTINUE
        WRITE(TERM, 510) INLNCT.INLNCT
510 FORMAT("OLINES WRITTEN =".16."; LAST LINE NUMBER =".16)
        STOP
600 CONTINUE
       STUP 600
700 CONTINUE
STUP 700
                       END
                                          END OF DOEC UPRINT PROGRAM
                                                                                                                                                                      LINES PRINTED=
Э
```

```
START OF DOZO JPRINT PROGRAM
                                                            DSNAME=D0031.CCDELN.FORT
          SUBRUUTINE CODELN(LENGTH. POLAR, COELCT, CODATA)
C
         C INITIALIZE MAKE UP CODE. MAKE UP CODE LENGTH
          40002=0
          MFE 42=0
CCC
      CHECK INPUTS
          IF (POLAR.LT.1.GR.PSLAR.GT.2) CALL EXIT IF (LENGTH.LT.U.GR.LENGTH.GT.1708) CALL EXIT
C
          IF (LENGTH .LE.63) GD TO 10
0000
      CALCULATE MAKE UP CODE INDEX. CODE, LENGTH AND WRITE TO CODE LINE
         INDL(=LENGTH/64+64
MCCDE=CODE(3.INDEX.POLER)
MLENG=CODE(1.INDEX.POLER)
CALL MI28(MCODE.COBUF.CDELCT+1.MLENG)
CDELCT=COELCT+MLENG
CDDATA=CODATA+MLENG
0000
       CALCULATE TEXAINATING CODE INDEX. CODE. LENGTH AND ADD TO CODE LINE
    10 CUNTINUE
INDIX=MUD (LENGTH, 64)+1
TCGDE=CDDE(3, INDEX, POLAR)
TLENG=CUDE(1, INDEX, POLAR)
CALL MISS(TCDDE, COBUF, CDELCT+1, TLENG)
CO SLOT=CDELCT+TLENG
CDDATA=CDDATA+TLENG
C
          NACTER
C M B
```

```
SUBROUTINE STATS (LENGTH. INLNCT. DIAG)
C
       INTEGER MTT (5). ITT (2.5). LENGTH (INLNCT)
REAL STT (2.5). SUM. SUM SQ
LOGICAL DIAG
     ****************** FILE DEFINITIONS *************
C*
       COM40N/FILES/TERM.LPFIL.PELFIL.OTFIL.EFFIL
C
       DATA MTT/0,24,48,96,192/
     -00 300 I=1.5
ITT(1.1)=10000
ITT(2.1)=0
       SUM=0.
       SUMSQ=0.
       DO 100 J=1. INLNCT
000
     FIND FILLED LINE LENGTH
   LEN=MAXO(LENGTH(J).MTT(I))
IF(DIAG) #RITE(TERM.50) LEN
50 FORMAT(I8)
CCC
     FIND MINIMUM LINE LENGTH
                                                                     UNCLASSIFIED
                                                                     UNCLASSIFIED
       ITT(1.1) = MINO(LEN.ITT(1.1))
CCC
    FIND MAXIMUM LINE LENGTH
       ITT(2,I) = MAXO(LEN,ITT(2,I))
CCC
     FIND SUM OF LENGTHS
       SU 4 SUM+FLOAT (LEN)
       SU 4S J= SUM SQ+ (FLOAT (LEN)) **2
  100 CONTINUE
CCC
     FIND SAMPLE MEAN AND STANDARD DEVIATION
       STT(1,1)=SUM/FLOAT(INLNCT)
       STT(2.1)=SQRT((SUMSQ-(SUM**2)/FLCAT(INLNCT))/FLCAT(INLNCT-1))
  300 CONTINUE
       WRITE(LPFIL.400)(ITT(1.1).1=1.5)
  400 FORMAT (
  #* CODED LINE*/

** LENGTH 0 MS

** STATISTICS:*//

** MINIMUM*.10X.5(18)//)

WRITE(LPFIL.410)(ITT(2.1).I=1.5)

410 FORMAT(
                                        MINIMUM TRANSMISSION TIME (4800 RPS) 1/
                                                  5 MS
                                                           10 MS
                                                                     23 MS
                                                                              40 MS 1/
       MAXIMUM*,10X.5(18)//)
WRITE(LPFIL.420)(STT(1.1).1=1.5)
      * *
   420 FORMAT (
   ** SA.APLE MEAN*.9X.5(F8.2)//)
WRITE(LPFIL.430)(STT(2.1).1=1.5)
430 FORMAT(
            STANDARD DEVIATION . 2x.5(F8.2))
C
       RE TURN
       END
             END UF DEEC UPRINT PROGRAM LINES PRINTED=
0
```

```
BLICK DATA
C
       IMPLICIT INTEGER (A-Z)
C
       CJ4MJN /G328IT/KIBIT(32),KZ8IT(32),LI8IT(32),LZ8IT(32)
C
       DATA KIBIT Z
                    Z30700000, Z40000000, Z20000000, Z100000000.
                    Z08000000.Z04000000.Z02000000.Z01000000.
Z0800000.Z0400000.Z0200000.Z01000000.
Z0800000.Z00040000.Z0002000.Z00010000.
Z09080000.Z0004000.Z0002000.Z000010000.
     SSE
     5
     દ
                    Z0000000.Z00000406.Z00000200.Z00000100.
                    Z00000080, Z00000040, Z00000020, Z00000010
                    Z30000003, Z3000004, Z00000002, Z00000001/
C
       V TIESH ATAC
                    Z7FFFFFFF, Z3FFFFFFF, ZDFFFFFFF, ZEFFFFFFF,
     00000000
                    ZFFF7FFF.ZFFFBFFF.ZFFFDFFF.ZFFFEFFF.ZFFFEFFF.ZFFFFFFF.ZFFFFEFF.
                    ZFFFFF7FF, ZFFFFBFF, ZFFFFFDFF, ZFFFFFEFF,
                    ZFFFFFF7F, ZFFFFFBF, ZFFFFFFCF, ZFFFFFFEF
                    ZFFFFFFF7,ZFFFFFFB,ZFFFFFFD,ZFFFFFFFE/
       DATA LIBIT /
                    Z80J000J0.ZC000J000.ZE00000J0.ZF0000000.
     દ
                    ZF300000.2FC000000.ZFE000000.ZFF0000CC.ZFFB00000.ZFFC00000.ZFFE00000.ZFFF60000.ZFFF60000.ZFFFE0000.ZFFFE0000.ZFFFE0000.
     3000000
                    ٤
                    ZFFFFFFFS, ZFFFFFFFC, ZFFFFFFE, ZFFFFFFFF/
C
       DATA LZGIT /
Z7FFFFFFF, Z3FFFFFFF, Z1FFFFFFF, Z0FFFFFFF,
                    500000
                    20000007,200000003,20000001,20000000/
```

END

```
START OF DOEC UPRINT PROGRAM .
                                         DSNAME=D0031.I48.FCRT
C14B
        INTEGER FUNCTION 148(18A, J8, N8)
IMPLICIT INTEGER (A-Z)
DIMENSION 18A(2)
C
C******* I4B RETURNS AN INTEGER VALUE FOR THE BIT STRING
C STARTING AT THE JB-TH BIT OF IBA
C AND CONSISTING OF NB BITS.
C
   ***** L48ELED COMMON /G328IT/ *****
        CJM4JN /G328IT/MASK(32),CCMASK(32),LIBIT(32),LZBIT(32)
INTEGER MASK,COMASK,LIBIT,LZBIT
IF(NB-1) 10.30.20
       STOP 10
CONTINUE
   20
        3-RN+G C=6H NC
        (SE. BR) ON IN = TEN
        JRE=JRH8/32+1
        JR 3=MOD(JRH3, 32)+1
N3R=MINO(NBT, JRB)
        JI 4= 32-NBR
COO
     SHIFT RIGHT 32-JRB BITS AND PUT IN ZERGS ON LEFT
        J=IdA(JRE)
        K= 32-JR3
        143=LAND(LZBIT(JIM),SHFTR(J,K))
     CALCULATE NUMBER OF BITS REMAINING IN LEFT PORTION IF ANY
        N37=N3T-N3R
IF (N3R .LE.O) RETURN
     IF LEFT PORTION EXISTS. SHIFT LEFT PORTION AND PORT WITH RIGHT PORTION
                                   SHIFT LEFT TO LINE UP WITH RIGHT
        J=LAND(IBA(JRE-1), LZBIT(32-NBR))
        K=32-JIM
148=LOR(148,SHFTL(J,K))
        RETURN
     BIT STRING HAS ONLY ONE BIT
        CONTINUE
        14d=)
        JBINJ= (JB-1)/32+1
MSKI ND=JB-(JBIND-1) #32
        IF (LAND(MASK(MSKIND). IBA(JBIND)).EQ.MASK(MSKIND)) 148=1
        FETURN
C
        END
```

_	SUBROUTINE <u>ERRMES(</u> PELBUF. OTBUF. PELMAX. VRE S. ERRCNT)
•	IMPLICIT IN TEGER (A-Z) + REAL ESP
C	COMMON /G32BIT/MASK(32).COMASK(32).LIBIT(32).LZBIT(32)
C C	
c	CCMMON/FILES/TERM.LPFIL.PELFIL.OTFIL.ERFIL
	OIMENSION PELBUF(60), OTBUF(60) COMMON/LOGIC/SEARCH, DIAG LOGICAL SEARCH, DIAG
Č.	************************************
C _	REWIND PELFIL REWIND OTFIL
	ERROR=0 OTELW=(PELMAX+32-1)/32 OTLNCT=0
ç	READ AN ERROR FREE LINE
_	100 CONTINUE READ(POLFIL.END=600.ERR=800) INLANO.INELCT.PELBUF IF(MOD(INLANO-1.VRES).NE.0) GO TO 100
כ כי	READ ANTERROR-CORRUPTED LINE
	200 CONTINUE READ(OTFIL, END=500, ERR=800) OTLNNO, OTELCT, OTBUF OTLNCT=OTLNCT+1
	UNCLASSIFIED

```
300 CONTINUE
      COUNT DIFFERENCES BETWEEN TRANSMITTED AND RECEIVED LINES
         DC 450 I=1.JTELW
IF(OTBUF(I).EQ.PELBUF(I)) GD TO 450
IF(.NOT.DIAG) GD TO 420
WRITE(TERM,413) INLNNO.DTLNNG.I.PELBUF(I).OTBUF(I)
         FORMAT (318,2212)
CONTINUE
DO 440 J=1,32
IF(14H(3TBUF(1),J,1).NE.I4B(PELBUF(I),J,1)) ERROR=ERROR+1
   420
   440 CONTINUE
         CONTINUE
          IF (OTLNNO-INLNNO) 200.100,580
CCC
      ERROR LINE NUMBER GREATER THAN GOOD LINE NUMBER; COUNT DIFFERENCES BETWEEN GOOD AND ALL WHITE LINE
   500 CONTINUE
         00 350 I=1,0TELW
IF(PELBUF(I).EQ.0) GU TO 550
IF(.NOT.DIAG) GD TO 520
                                                      .
          WRITE(TERM,410) INLNNJ.OTLNNO.I.PELBUF(I).OTBUF(I)
   520 CONTINUE
          DO 540 J=1.32
If([48(]ELBUF([],J.1).NE.0) ERRCR=ERRCF+1
   540 CONTINUE
550 CONTINUE
   580 READ(PELFIL.END=500.ERR=800) TINLNNO.INELCT.PELBUF
          IF (MOD (INLNNO-1. VRES) . NE. 0) GO TC 580
C
          GO TO 300
c
      CALCULATE ERROR SENSITIVITY FACTOR ...
   600 CONTINUE
          ESF=0.
          IF (ERRCNT.LE.O) GD TO 650
ESF=FLOAT (ERROR)/FLOAT (ERRCNT)
   650 CONTINUE
   WRITE(LPFIL.700) ERROR.ERRCNT.ESF.OTLNCT

700 FORMAT('ONUMBER OF INCORRECT PELS #',110/

# 'ONUMBER OF BITS IN ERROR TRANSMITTED #',110/

* 'OERROR SENSITIVITY FACTOR #',F12.4/

* 'OTOTAL NUMBER OF DUTPUT LINES PROCESSED # '.18)
C
          RETURN
   800 CONTINUE
STOP 830
          END
```

- (IM M 414	IN & FENST	81 -	· · · · · · · · · · · · · · · · · · ·	DATE -	79176	14/06
•	Ċ	PROCEE	WALTADA?			
35.71		I MPL IC IT	INTEGER (1-2) PELDLE (C3)CTOUE (1500)			
	er entre i recommenda en entre en entre en en entre	CATA PEL	MAY . PELF IL . OT FIL . TERM/I	720.1.2.5/	å : vi	interpreter information with Missingle.
	Ž.a.a.a.v	r-myr yryrydd	нишилцика _н кала. Лебін ірп	OGRAM 444.144#	***	******
1114	c	INLUCTED	and the second s	and the second second		24.1.1
3735 3376		CONTINUE	=1,60			
30.34			FIL, EUD=5 ** , ERR=6000 IN	LNNU. INELCT .P	elbûř	THE SERVICE OF THE OWNER OF THE OWNER OF THE OWNER OWNER.
9519	· c depth y c		NLNGT+1 &LOUF+1+1-)	• • •	3. ·	
2015		PCLAR#IC J#1	on de de la constitución de de designación de designación de	•		
0517	a to the day was to the		=1.PELMAX	•		
3215	.,,	PEL = 1.40 ()	PELDLE . L . L . L . L . L . L . L . L . L .	ا يوره مسد		, secular expension
7017 3518		018UF(J)	#G(JA)	• •		
1022		RUN=1	C(FCLAR+1,2)			
0022 0022		CLNT INUE	5 · / · · · · · · · · · · · · · · · · ·	•		
- 323 - 3024		RUN=EUN+		aan Japan Maraagaan ka salka sooga / s. sa . sa	All makes to the Albania of the Marketine of the Marketin	e qui regionale de la terrar de des la reconstant a
1025		0 TBUF (J)	> RUN			
3026 3.27	e-and-early to the	IF=25		* *	·	
202 a 30 5 8		T) TAPACE	FIL.311) IC.J			
	316	J1=J 1F(_J_oG_T_c	25 <u>0) GO TO 315</u>	and the second of the second o		a appropriately and the second
0033		J131=J+1)• J1C1)		
JC34 -	315	GO TO 46 CENTINUE		***		
3136 0027	and agreement to the contract of		FIL.3))) (OTBUF(K).K=10).IF)		1 1 1
6.57.6		IF=IF+25 J=J-251	ɔ			
. 14 1		IF(J.E).	2) GC TC 400	grande from the state of the st	Adv. 1. 4 101-101-11-11-1-1-1-1-1-1-1-1-1-1-1-1-1	
72.41 72.42	40°	CONT HUE	· vinda na kisalaku kati u dalamatan manara anta kansaki wi vinda na kisali			4.5
2,42		FURMAT(5	FM.417) IC, JI X.14.5X.15)			
2044		FORMAT(2	RM;"#27)""("OTBOF(K);K=1".J X.2")([4.2X]))1)		•
1 10	5	CONTINUE		a territoria del mante del como estra en la como estra en	apage, a series y decima	ing and an analysis of the second
3017 3643		INTAKHCA	RM.51') INLNCT.INLNNO OLINES-WRITTEN-F1.16.'!	LAST LINE NO	MECR WIN	10)
2045	6.10	CCHTINUE				
07.51° 105.2	Committee of the second section of the second	~~\$TJP_67 <i>*</i> *	B and Co. and against the company of	. •••		*
				CONTRACTOR OF THE CONTRACTOR O	and control of a special who be	a a market i new ye in e mana
	A Company of the Comp	er dan er eredente	and the second states of the second states and the second states and the second	a mana o some property.		N
-				ه خطف ناست است. خطف ناست میشدند.		٠′0 <i>K</i> ,
					10	
• • • • • • • • • • • • • • • • • • •				Best AV	:190	and the second s
to your day of the second second	. I we so dispenses the second section in the second	a. 1884. 1894. 1784. 1784. 1894.	entidoscophido de qualita de la compansión		Sir.	w
مه در دور در خوره (د	un in de design wild ou de gans legion débite	Market and the second s		was the second		
				dest		
-		ادل و داناه بدانها الماسي اطران د		The second section of the second second	and the same of the same	a entrent see the seed of the seed
olic contage i — prof. Minus	and the second s	erigin engler i en slore objected desperius :		unintrop des des introgénia destruita () (no. 1901) au 1900.	. n. with the site of	May 12 3 for 1 2

·C- 9 ·

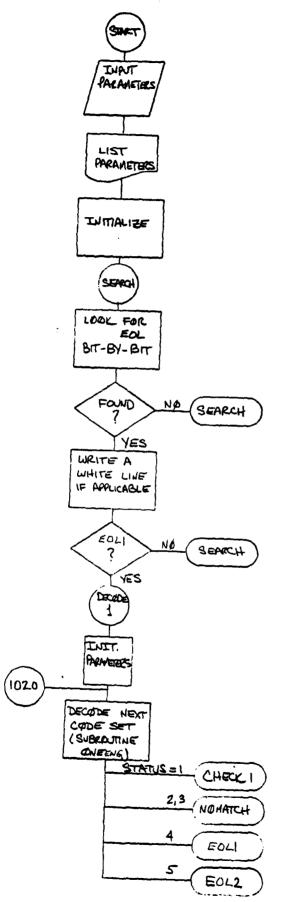
UNCL4331F1ED

51 C	TART	DF DCEC UPRINT PROGRAM DSNAME=D0031.CONVERT.FORT PROGRAM CONVERT
0000		HIS PROGRAM CONVERTS BINARY FORMAT USED BY COMPRESSION GURITHMS TO THE FOLLOWING BINARY FORMAT:
0000		1728 BITS (216 BYTES) PER RECORD;
Š	E.	ACH LINE OF 1728 PELS BECOMES ONE RECORD
		IMPLICIT INTEGER(A-Z) INTEGER PELBUF(60).OTBUF(54) EQUIVALENCE (PELBUF(1).CTBUF(1)) INLNCT=0
_	100	READ(1,END=500,ERR=600) INLNNO,INELCT,PELBUF INLNCT=INLNCT+1 WRITE(2,ERR=700) OTBUF GO TO 100
·	500	CONTINUE WRITE(5.510) INLNCT.INLNNC
	510	FORMAT(LINES WRITTEN = 1,16.1; LAST LINE NUMBER = 1,16)
		STOP CONTINUE STOP 500
	703	STOP 700 E N D
0		END OF DOES UPRINT PROGRAM LINES PRINTED= 26

APPENDIX D

FLOW CHART

BRITISH POST OFFICE



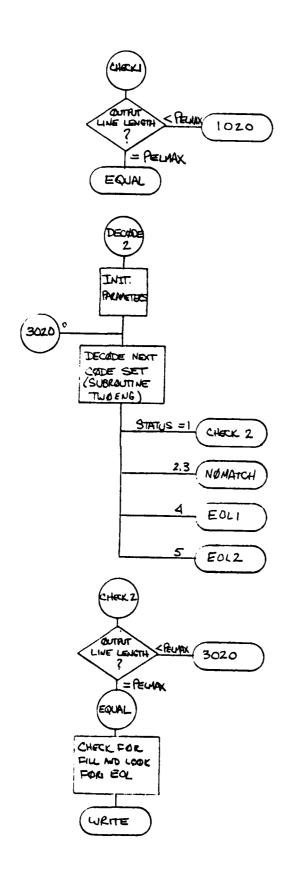
ENGLSH!

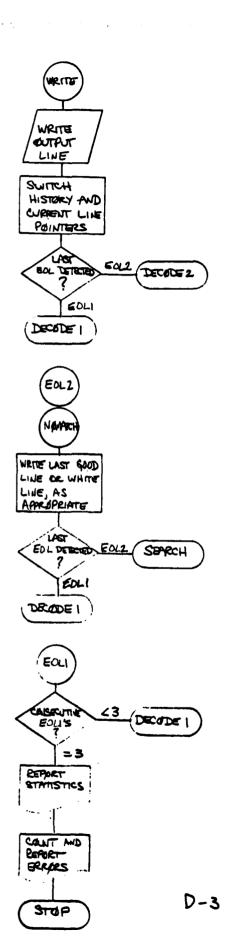
OUTPUT LINE LENGTH & PELHAX

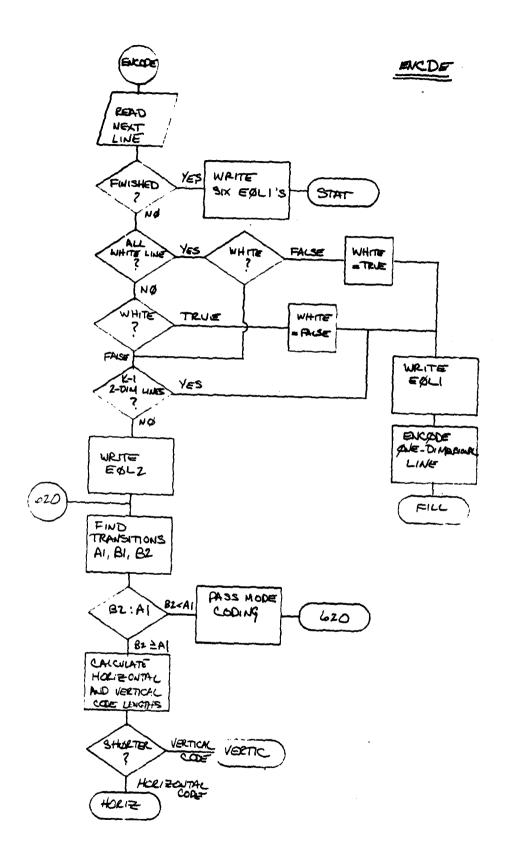
OUTRIT LINE TOO LONG OR NO MATCH FOUND IN CODE TABLE

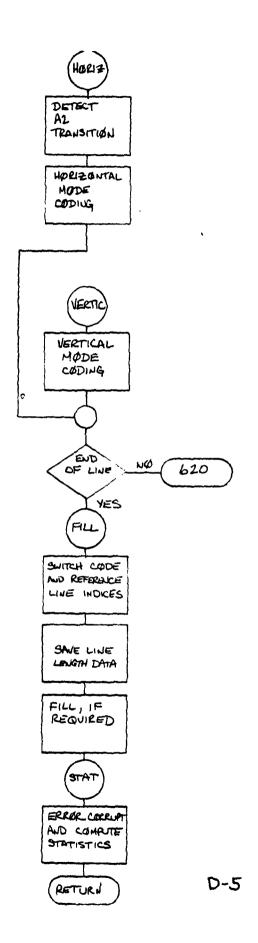
PREMATURE EOL DETECTED

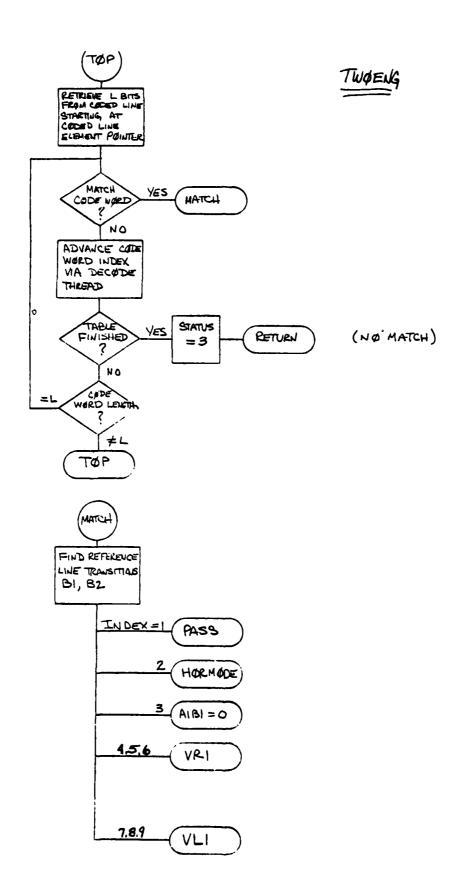
D-1

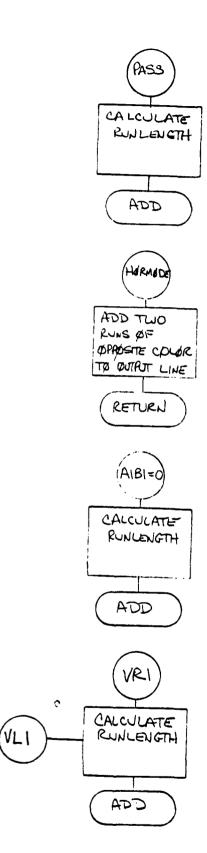












APPENDIX E

CODE LISTING
BRITISH POST OFFICE

START OF COEC JORINT PROGRAM DENAMERNOUZE. ENGLISH. FORT	
C PROGRAM ENGLISH	000000
IMP. ICIT INTEGER(A=Z)	200000
REAL CF3, CF4, ERRATE	000000
C++++++ LABELED COMMON /G32BIT/ ++++++	00000€
COMMON /J3281T/MASK(32).COMASK(32).L181T(32).L231T(32)	000000
INTEGER WASK, COMASK, LIBIT, LZBIT	000000
COMMON DUES (OST DUES (A.D. CODUS (2001) (2001)	000000
COMMCN/BUFF/PELBUF(60,2),CDBUF(240),OTEUF(6),2), * STEBUF(240), STAT(3000)	100000
COMMCN/HUFF /CODE(3,92,2),CCDERD(3,11)	00000:
COMMON/ER AY/ERRORS (2500)	000001
	000001
CCMMCN/FILES/TERM.LPFIL.PELFIL.OTFIL.ERFIL	000001
	00000
C*************************************	100000
COMMEN / I VAR / PEL MAX . VRES . EPHASE . CMPMAX . ERRMOD . LINMAX . K	00000
COMMCN/PY AR/IN_NND.OTELW.INELP.CDELP.UT=LP.CDELW.	00000
COLLETY INCLETY TO DESCRIPTION OF YEAR OF YEAR INVOICE CHIEF OF THE PROPERTY O	000001
* ERRCNT, INLNCT, CONSEC. ONECNT, LNNOBF, KCNT. * INCOD, INREF, CTCOD, OTREF, STFBIT	00000
COMMONZIC HARZOD . II . MM . TT . NN . YY	00000:
COMMEN/LOGIC/SEARCH, DIAG, SYNC, WRITE, ZERD, LEFT, CHCUL, DNE, WHITE	00000
LOGICAL SEARCH DIAG SYNC WRITE LEFT CHCOL UNE WHITE	0000
C READ INPUT PARAMETERS	00000.
	000001
100 FORMAT('SPARAMETERS: INPUT(=I), OR DEFAULT(=D)?') READ(5,110,5RR=90) INSW	00000.
IIO FLEMAT (AL)	00000
IF (INSW.EQ.DD) GO TO 315	00000.
IF (INSWeNE - II) GO TO 90	00000
C C READ DIAGNOSTIC SWITCH	000000
The state of the s	00000.
114 WRITE(6,115)	00000.
115 FORMAT('\$DIAGNOSTIC PRINTOUT? (Y OR N): ') READ(5.110) INSW	00000
IF(INSW. 20. YY) GO TO 116	00000
IF (INSW-=Q+NN) GO TO 120	00000
GO TO 114 116 CONTINUE	00000
01 AG = 1 TQ E 5	-00000
C	00000
C READ MAXIMUM NUMBER OF PELS PER LINE	00000
120 CONTINUE	00000
*RITE(6.) 30)	.00000
130 FORMAT ("SENTER MAXIMUM NUMBER OF PELS PER LINE: ")	00000
REAC(5,140, ERR=120) PELMAX	-00000
IF (PELMAX .SE.1.AND.PELMAX.LE.1728) GO TO 160	00000:
WRITE(6,150) PELMAX	00000
GO TO 120	00000
<u> </u>	00000
C READ VERTICAL SAMPLING	00000
C - 160 - CONTIMUE	-00000
WRITE(6.170)	00000
170 FORMAT('SENTER VERTICAL SAMPLING: ')	00000
REAC(5.180. ERR=160) VRES 180 FORMAT(12)	00000
TE (VEES-35-1-AND-VEES-1F-10) GO TO 190	00000
WRITE(6,150), VRES	00000
GO TO 163	00000 -0000
C READ PARAMETER K	00000
<u>C</u>	00000
190 CONTINUE	00000
WRITE(6,192) 192 FORMAT(1554TER PARAMETER K: 1)	20020
READ(5.140.ERR#190) K	00000
IF(K.GE.1.AND.K.LE.3000) GD TD 200	00000
GO 10 193	-00000
C	00000
C READ ERROR PATTERN PHASE	20002
C	00000

200 CUNTINUE WRITE(6.210)	000001
210 FORMATIC SENSE PATTERN PHASE: 1)	00000
READ(5.220.ERR=200) EPHASE	00000
220 FD FMAT (11)	00000
WRITE(6,150) SPHASE	00000
GO TO 200	000001
	00000
C READ MINIMUM COMPRESSED LINE LENGTH	00000
240 CCNTINUE	00000
WRITE(6, 250)	000009
READ(5,140,ERR≈240) CMPMAX 1f(CMPMAX.JE.0.AND.CMPMAX.LE.1728) GD TO 320	00000
WRITE(6.150) CMPMAX	00000
GO TO 240	000004
C READ NUMBER OF SCAN LINES TO BE PROCESSED	00001
320 CONTINUE	00001
WRITE(CVJ 30)	- 00001
330 FORMAT('SNUMBER OF SCAN LINES TO BE PROCESSED=? ')	00001+ 00001
READ(5,140,ERR=320) LINMAX IF(LINMAX.GE.LAND.LINMAX.LE.3000) GC TO 280	00001
WRITE(6.150) LINMAX	00001
GO 10 320	00001
C READ ERROR MODE	00001
280 CONTINUE	00001
WRITE(6.290) 290 FORMAT('SERROR MODE=? (M=MANUAL.TETAPE.N=ND ERRORS)')	00001
READ (5 .110 . ERR=280) ERRMCD	00001
	00001
IF(EFFMOD.EQ.TT) GO TO 315 IF(EFRMOD.NE.NN) GO TO 280	00001 00001
C	00001
C READ ERROR LOCATIONS	00001
300 CENTINLE	00001
FRRL IM=L	00001
305 REAC(5,140) ERRORS(ERRLIM)	00001
IF(ERRORS(ERRLIM).EQ.9999) GD TO 310	
GD TO 335	00001
310 CONTINUE ERRLIM=ERRLIM-1	00001
GO TD 350	00001
	00001
C READ ERROR TAPE FILE AND OPEN C	00001 00001
- 315 - CENT INUE	
C	00001
ERRLIM#1	00001
READ(3,315,END=317) ERFORS(ERRLIM) ERRLIM=ERRLIM+1	00001
316 REAC(3.318.5ND=317) ERRORS(ERRLIM)	00001
318 FORMAT(I16)	00001
ERRORS (ERRLIM)=ERRORS (ERRLIM)+ERRORS (ERRLIM-1)	00001
GD TO 216	00001
317 ERRLIM=ERRLIM-1	00001
350 CONT INUE	10000
<u> </u>	00001
360 CONTINUE	00001
C WRITE INPUT PARAMETERS	00001
WRITE(6.400) PELMAX, VRES, K. EPHASE, CMPMAX, LINMAX	00001
400 FORMAT ('I INPUT PARAMETERS: '/	00001
* 'OMAXIMUM NUMBER OF PELS PER LINE='.157 * 'OVERTICAL SAMPLING: N='.14/	00001
1 OPARAMETER K = 14/	00001
* ** ** ** ** ** ** ** ** ** ** ** ** *	00001
# 'J MI NI MUM COMPRESSED LINE LENGTH #', 14, ' BITS'/	00001
IF (ERRMOD. EQ.NN) WRITE(6.410)	00001
410 FORMAT ('0:NO ERROR'S I VSERTED') IF(ERRMID .EQ.MM) WRITE(6.140) (ERROR'S(I).I=I.ERRLIM)	00001
IF(ERRMOD.=EG.MM) WRITE(6.140) (ERRORS(I).I=1.ERRLIM) IF(ERRMOD.=EG.TT) WRITE(6.420) ERRLIM	00001
TITE CONTRACTOR OF THE LAND CONTRACTOR	20001

C*****		000
	ITIALI ZE	000
C	SCOTI - A	000
	TCDE L= 0	000
	ERRPNT=[000
	FRENT=0	000
	INLNCT =0	000
	ERRCFE =EPHASE 1 024 CDEL CT=32	_000
	CT all P= 1	000
		-000
	CCNS EC=1	000
	I NREF = 1 IN COD= 2	000
	07 REF=1	000
	TIC00= 2	000
	WHITE= FALSE.	000
	KCNT=1	000
•	CO 800 I=1.240	000
	STFBUF(I)=0	000
	CCPUF(I)=0	000
	CONTINUE DO 850 1= 1.60	000
	CT BUF(I,) TR EF)=0	000
	THUF(1,3 TC00)=0	000
	PEL SUF (IV IVREF) = 0	
	PELBUF(I,INCQ))=0	000
	CONTINUE SEARCHE-TRUE	- 000
	SYNC=-FAL SE.	000
	WE ITE= JEAL SE	000
SE	AS CU MODE A 4 0 CW S CO D SOLVE DATE OF THE COLUMN TO THE	000
3E.	ARCH MODE: LOCK FOR EOLI BIT-BY-BIT	000 000
	CONTINUE	000
	CALL GETLE(13.MODE.LBITS.L)	000
	GC TC (910.930.930.920).MCDE	000
	STOP 900 CONTINUE	000
	·	000
	L NCT FOUNCE POINTER AND TRY BAIN	000
	COEL 5-60 14 044	-000
	CDELF=CDELP+1 GD TD 900	000
	ČČNÝ INÚŠ	000
	STOP 920	000
	CENT INUE	_000
EO	_ FOUND	000
		-000
	SEARCH=.= AL SE.	000
	CD SLP = CDELP +L	000
	IF (ARITE) GC TC 935 MRITE= •TRUE•	000
	60 12 363	_000
935	CONTINUE	000
		000
SE	T CUTPUT DECODE LINE TO 0 AND WRITE OUT	- 900
	CT 8ÚF (1. 1 COD) =0	000
950	CHINE	000
	WRITE(2) OTLNHO.PELMAX.(OTBUF(1.OTCOD).I=1.60)	000
	CTLNAC=LUMDBE CONTINUE	000
	IF (MCD2-2)965, 1000, 900	000
965 -	STOP 965	-000
	CONT IN US	000
PE	REGRM ONE-DIMENSIONAL DECODE OF A COMPLETE LINE	000
FI	RECEM UNITIONAL DECODE OF A COMPLETE LINE RST.SET DUTPUT BUFFER TO WHITE	000
	NLY BLACK RUNS WILL BE INSERTED)	_000
		000
	00 1010 I =1 .60	000
	DT BUP (1 + 3 TC 3 D) = 0	000
1010		000
	IN CEX = 3	000
	CGLCF=1	000

UTELPET	000024
c states.	000024
1000 CONTINUE	000024
CALL ONE SIG (INDEX, COLOR, STATUS, L)	000024
GO TO (1030,1070,1070,1035,1040),STATUS	000025
STOP 1000	000025
C	000025
RUN ADDED! CHECK LENGTH OF DUTPUT LINE	000025
1030 CONTINUE	000025
ON E= .TRUE .	000025
IF(OTELP-1-PELMAX) 1031.1032.1050	000025
IF (CHCOL) COLOR=MOD (CDLOR+2,2)+1	
INDEXES	000020
GD TO 1020	000028
3000 CONTINUE	00002t
C PERFORM TWU-DIMENSIONAL DECODE	000026
C C	000026
C FIRST-SET DUTPUT BUFFER TO WHITE C (DNLY BLACK RUNS WILL BE INSERTED)	
C FIRST-SET DUTPUT BUFFER TO WHITE	00002 <i>t</i> 00002 <i>t</i>
C (CNLY BLACK RUNS WILL BE INSERTED)	00002
DO 301 C I =1 ,60	00002
OI BUE(1.3 1003)=0	000027
3010 CCNTINUE	00002 ⁻ 000027
1N85x+3	
ČÓĽĆR=Ĭ	00005.
OTELP= 1	000027
3020 CCNTINLE	000027
CALL TWOENS (INDEX COLOR STATUS L)	000021
GO TO (33 30 .1070.1070, 1035, 1040) . STATUS	00002
1 2 3 4 5	00002i
C 37 CP 3000	00002
C RUN ADDED; LOOK FOR NEXT RUN	00002
	00002
3030 CCNTINUE	00002i
IF(CTELP-1-PELMAX) 3031,1032,1050	00002
3031 CONTINUS	000029
1F (C+CL) COLOR=MOD(COLOR+2v2)+1	000029
INCEX=3 GD TO 3020	00002
	00002
C ,	00002
C LINE LENGTHEREL MAX: CHECK FOR FILL AND LOOK FOR EGL	00002
1032 CONTINUE	00002:
20AC	
1033 CONTINE	00003i 00003i
ZERO=ZERO +1 CALL GETLE(1,MODE,LBITS,L)	00003
C	00003:
GO TC (13 34-1050-1050-1050)-MODE	00003; 00003(
C CHECK FOR FILL	000030
1034 CONTINUE	000030
C	00003
CDELF=CD2LF+L IF(LBITS.EQ.0) GD TO 1033	00003
IF (ZER Cal Falo) GO TO 1070	00003
C ECL FCUNC; CHECK TYPE	90003.
C ECL FCUNC; CHECK TYPE	00003:
CALL GET_E(1,MODE,LBITS,L)	00003
IF (L8175. E2.1) MODE=2	00003
IF(LBITS.EQ.O) MDDE=3 GD TO (1070,1060,1060,1080),MDDE	00003
	00003
C PREMATURE EOL DETECTED	00003:
<u> </u>	00003
C PREMATURE EQL DETECTED C ECL1 DETECTED	
	00003.
1035 CONTINUE	00003:
CDELP=CDE L9+L	00003;

STATUS=4	00003.
IF (OTELP.LE.I) CONSEC = CONSEC + I	00003
IE (CCNSEC-2)1080-1000-2000	00003.
C EDL2 DETECTED	00003.
	- 0000 3
1040 CONTINUE	00003:
CDELP=CDEL.3+L	00003.
31A1U3=3	00003:
<u>GO 1C 1080</u>	90003.
C	00003.
C PROBLEMS PROBLE 4S	00003
1050 STOP 1050	000034
C THE THE PARTY OF	00003
C LINE LENGTA CORRECT. EUL DETECTED PROPERLY: WRITE OUTPUT LINE	00003
1 060 CENTINE	- 20003.
CDELP# CDELP ML	000034
WRITE(2)JTL NOO .PELMAX . (CTRUF(I.OTCOD) . I= 1.00)	00003
CONSEC=1	000039
IF (CNE) SYIC=.TRUE.	00003:
	000071
CTREF=OTCOD OTCOD= IEAP	00003
IF(MDDE-23-2) GO TO 1000	00003:
GD TC 3000	00003:
C LINE TCO LUNG OR NO MATCH	000035
	00003
C TOTO CONTINUE	000331
WRITE=-FALSE.	000034
C LINE SHORT	00003c
C 1680 CONT INU!	0003 :
IF (.NOT.SY IC) GO TO 1090	000030
	000034
C WRITE LAST JOOD LINE	00003
C	000031
WRITE(2) DILNNO.PELMAX.(GTRUF(1.CTREF).I=1.60)	00003
SYNC=.FALSE. GO TC 1110	000037
1090 CONT INUE	00003i
	000037
C WRITE A WHITE LINE	00003
	00003
DO 1100 I = 1, 60 1100 OTEUF(1.JTCOD)=0	00003
WRITE(2) JTLNNO.PELMAX.(CTBUF(1.CTCCD). [=1.60)	00003
1110 CTLNNG=LNNJBF	000038
IF (STATUS +2-9+4) 60 TO 1000	000031
SEARCH = TRUE	00003: 00003:
GO TO 900	00003
C END OF MESSAGE	00003:
<u> </u>	00003.
2000 CONTINUE	00003
WRITE(6,2010) CONSEC	00003:
2010 POPMATION OF MESSAGE DEFLOTED (*, 12, * EJE * 5) *)	000035
C REPORT COMPRESSION FACTOR, ERROR SENSITIVITY FACTURIBLE ERROR RAT	E 00003
C REPORT COMPRESSION FACTOR, ERROR SENSITIVITY FACTURABLE ERROR RAT	00003
CODATE SE MATICODONTA /EL MATITODELA	000035
WRITE(6.2020) TCDEL TCDATA STERIT INLNCT FRATE	
2020 FORMAT('UTOTAL NUMBER OF CODED BITS = '.18/	27 0000
+ 'DIGITAL NUMBER OF CODED DATA RITS = '.13/	
+ '3 BIT =FROR RATE = '.G14.6)	00003
CALL STATS(STAT, IN_NCT,DIAG) CFJ=FLCAT(P=LMAX)RFLDAT(INLNCT)/FLDAT(ICCEL)	00004
CF 4= FL CAT (2 =LMA X) #FL DAT (INLICT) /FL GAT (TCDATA)	000040
C C	
2030 FORMAT (") C) APRESSION FACTOR FOR G3 MACHINE (C=3) = 1.F8.4/ 1) C) APRESSION FACTOR FOR G4 MACHINE (CF4) = 1.F8.4)	00004
JC3 PRESSION FACTOR FOR G4 MACHINE (CF4) = 0.68.41	00004
CALL ERRAES (PEL DIFOTBUFOPEL MAXOVRESOFRECOT)	000040
CALL CHAILS (PGL DIFFICTION FIRE MAXIMESTANCH)	

-		
	STOP STOP	00004
	SUBROUTINE GETLE(LBITS.MCDE.WRD.L) IMPLICIT INTEGER(A-Z)	00004
****	+++- <u>LABELEO-CO4MON / GOE BET/ -</u> +++++	-0000+
С	CCMMCN /5328[T/MASK(32),CUMASK(32),LIBIT(32),LZBIT(32)	00004
<u> </u>	INTEGER MASK, COMASK, LIBIT, LZBIT	00004
C	COMMENSAUFF SPELBUE(60.2). COBUF(240).CTBUF(60.2).	00004
	• STFBUF(240), STAT(3000) CCMMCN/HUFF/CDDE(3.92.2).CQDERD(3.11)	00004
	COMMON FRAY/ERRORS(2500)	00004 -0000
C****		00004
	COMMON /IV AR /P =LMAX , VRES , EPHASE , CMPMAX , ERRHOD , LINMAX , K COMMON / PV AR / I NL 'NO , OT L NNO , OT ELW , INELP , CDELP , DTELP , CDEL W ,	00004
	COELCT. INELCT. TCDATA TCDEL ERRPNT EURIFF ERRLING	00004
	# ERRCNT, INLNCT,CONSEC,DNECNT,LNNO3*,KCNI, # I'NCOD,INR3F,CTCOD,OTREF,STFBIT	00004
	- COMMON/IC MAR/DD-11-MM-TT-NN-VY - COMMON/LOGIC/SEARCH.DIAG.SYNC.WRITE.ZERO.LEFT.CHCGL.DNE.WHITE	-00004
	LOGICAL SEARCH.DIAG.SYNC.WRITE.ZERO.LEFT.CHCUL.UNE.WHITE	00004
	AAAAAAAAAAAAAAAAAAAAA BEGIN PROGRAM AAAAAAAAAAAAAAAAAAAAAAAAA	00004
	MO CE=A	00004
C R	ETRIEVE NEXT BIT FROM COBUF	00004
		-00004
L 00	CONTINUE	00004
Ē	NCOCE A NEW LINE IF NECESSARY	00004
C	JE(LBITS+CDFLP+1-LF-CDFLCT) GD TO 20C	00004
	1F(CDELCT-CDELP+1) 170.190.180	00004
	STOP 170	00004-
	STFBUF(1) =1 48(STFBUF,CDELP,CDELCT-CDELP+1)	00004
190	CONTINUE CDELP=32-(CCELCY-CDELP)	00004
200	CALL ENCJE	00004
-2.00	WRD=148(STFBUF.CDELP.LBITS)	00004
	L=LBITS - IP(L+LT+13) 62 13 250	00004
	IF(L.EQ.13.AND.WRD.EQ.CODERD(3,10)) GO TO 300	00004
250	IF(L.EG.13.AND.WRD.EG.COCERD(3.11)) GO TO 400	00004
	MOCE=1	00004
30.0	RETURN CONTINUE	00004
• • •	MOCE=2	00004
400	CONT INUE	00004
	MODE =3	00004
	RE TURN ENC	00004
<u> </u>	SUBROUTINE ENCDE	-00004
•	IMFLICIT INTEGER(A-Z)	00004
C***	*** LARELED COMMON /G328IT/ ******	00004
<u>c </u>	COMMON 75 J281 T/MASK(32), COMASK(32), LIBIT(32), LZBIT(32)	00004
_	INTEGER WASK, COMASK, LIBIT, LZBIT	00004
<u> </u>	COMMON /BJFF/PEL BJF (60.2) .CDBUF (240) .D TBUF (60.2) .	00004
	STEBUF(240). STAT(3000)	00004
	- COMMEN / IUF / CODE(3 y 92 y 2) y CODERD (3 y 1 1)	-00004
	************** FILE DEFINITIONS ***********	00004
ζ	COMMCN/FILES/TERM.LPFIL.PELFIL.OTFIL.ERFIL	00004
C		00004
C	++++++++++++++++++++++++++++++++++++++	00004
	-COMMONAIV ARAPELMAXIVES VEPHASE VEMPMAXVERRADO VEINMAXVE	-0000+
	COMMENZEY ARZINUMNO.OT LUNDO.OT ELW.INELP.CDELP.DELP.CDELW. COELCT.INELCT.TCDATA.TCDEL.ERRENT.ERREFF.ERRLIM.	00004
	ERR CNT. INLINCT.CONSEC.CNECHT.LNNJSF.KCNT.	00004
	* INCOD,INREF,CTCOD,CTREF,STFBIT	00004

	DMMON/ICHAR/DD,II,MM.TT.NN.YY DMMCN/LJGIC/SEARCH.DIAG.SYNC.WRITE.ZERO.LEFT.CHCDL.JME.WHITE OGICAL SEARCH.DIAG.SYNC.WRITE.ZERC.LEFT.CHCDL.LNE.WHITE	00004 00004
Ç•••••	**************************************	00004
	TIALIZE VARIABLES	90004
<u> </u>	CRY=KCY-I	00004
	COELCT =32	00005
	ORATAGO	.00005
	CO 50 1=2,240	00005
	DBUF(I)=0	00005 00005
50 C	ONTINUE	00005
C C REA C		00005
C ME	ID TROUT PICTURE FILE	00005
	ONT BUE	00005
	EAC(1 .EHD=120.ERR=500)	00005
	INLNNO.INELCT.(PELBUF(I.INCCD).I=1.60) IF(MED(INLNNO-1.VRES).NE.0) GD TO 100	00005 -00005
	F(INELCT.LT.PELMAX) CALL EXIT	00005
	NLNCT=INLNCT+1	00005
C LOA	O OLYOUT LINE MUMOED SHEEFS	00005
- 204	D OLTPUT LINE NUMBER BUFFER	00005
L	.NNCBF = INL 'INO	00005
	F (SEARCH) D TLNNO=LNNO3F	00005
	F(INLNNU-LE-LINMAX) GC TC 140	-90905 90005
•	FINEWOOLESCHMAN, OC 1C 140	00005
प्रत ा	TE SIX ECLITS	00005
	TO TAME	00005
	00 120 I=1.6	<u> </u>
-	ALL CCDING(10.0.0.0.CDELCT.CDDATA)	00005
130-6	CHINC	00005
	CO 125 I= 1.6	00005
	TFBUF (I) =COBUF(I)	00005
	50 TO 400	00005
<u> </u>		-00005
_ 140 (CONT IN UE	00005
72:	T FOR ALL WITTE LINE	-00005
c		00005
	NELW=(I 1ELCT+32-1)/32 D 145 I= 1, INELW	00005
	(F(PEL BUF(1, INCOD)) 146.145.146	00005
	ONTINE	.00005
c	in to all multip	00005
C LIN	NÉ IS ALL WHITE	00005 -00005
1	IF(WHITE) GO TO 147	00005
	HITE= TRUE .	00005
c '	50 YO 149	00005
Č LIM	E IS NOT ALL WHITE	00005
C		00005
	CONTINUE TO CONTI	00005 00005
	MHITE= FALSE	00005
	iQ TO 149	00005
C T: 6	ET EEC C METCHTINE OF THE LINES	00005
C	T FCR CJNSECUTIVE 2-JIM LINES	00005 20005
	EUNI INDE	00005
	F(KCNT) 149,149,600	00005
	CONTINUE	- 00005
c		00005
	-CIMENSID TAL CODING	00005
C WA	ITE CNE EDL1	00005
	CALL CCDENG(10.0.0.0.0.CDELCT.CDDATA)	00005
c `		00005
	PCL #R#1	-00005
C 155	T COLOR OF FIRST ELEMENT	00005
	ST COLUR OF FIRST ELEMENT	00005
	IF([48(PEL3UF(1.INCOD).1.1).EQ.0) GC TC 150	00005

C FIRST ELEMENT BLACK; ENCODE O-LENGTH WHITE RUN	00005 00005
CALL CODELN(0.1.COELCT,CDOATA)	00005 00005 00005
C CALCULATE RUN LENGTH AND ENCODE	00005
C 150 CONTINUE	00005
RUN≠ C	00005
CO 200 I= 1.2EL MAX PEL=148(2EL BUF(1, INCOD), 1,1)+1	00005
IF (PEL .EU . POLAR) GD TO 180	00005
IF(.NOT.DIAG) GD TO 170	00005
WR ITE(6,160) RUN, POLAR, CDELCT, CDDATA	00005
160 FCRMAY(413) 170 CONTINUE	00005 00005
- GUARI	00005
POLAR=MO) (POLAR+2.2)+1 GD TC 200	00005 00005
- 100 CONTINUE	
RUN≠RUN+1 200 CCNT INUE	00005 00005
CALL COSTITUTION, POLAR, COELET, CODATA)	00005
KC NT = K 	00005
WRITE(6.160) RUN.POLAR.CDELCT.CDDATA	00006
GO TO 210	00006
TWO-CIMENSIONAL CODING	00006
C 600 CONTINUS	00006
STFBIT=STFBIT+1	00006
C WRITE ONE EOL 2	00006
C WRITE ONE EOL 2	000060 00006
C SET AO TO LEFT EDGE-1 AND POLARITY=WHITE	0000b
	00000
A0 = 0 POL = C	00006
LEFT=.TRUS.	00006
C	00006
C	00006.
620 CCNTINUE 1= A0+1	00006
IF(I.GT.PELMAX) GO TO 640	00006.
PEL=148(PSL BUF(1.INCOD).I.1)	00006
IF(PEL.NE.POL) GO TO 640	00006
IF(1.LE.PEL MAX) GO TO 630	
640 CONTINUE	00006.
C Alsi	00006. 00006
C DETECT R1	00000
C	00006 00006
PELMI=143(PELBUF(I,INREF),A0,1) IF(LEFT) PELMI=0	00006 00006.
650 CONTINUE	30000
PEL=14B(PELBUF(1,1NREF).1.1) 	00000
660 CENTINUE	00006
PELM1=PE	00006
IF(I.LE.PELMAX) GD TD 650	00000
665 CONTINUE BI=I	00006·
GO TO 713	000064
IF (PEL·NE·POL) GO TO 690	00006
GC TC 660	00006
## ## ## ## ## ## ## ## ## ## ## ## ##	
FÖL±PEL	00006. 00006
C DETECT H2	.50000
o person de	00006:

E-8

I=01+1	000065
IF (Lagrapel MAX) GD TO 710	000065
700 CONTINUE	000065
PEL= 146(>= 8UF(1, INREF), 1, 1)	000065
1F (PEL VN2 4-> CL) GO TO 720	
I = I + 1	00006c
IF (I LE PEL MAX) SO TO 700	300006
Ma Carling	3900066
82=1 	000066
720 CONTINUE	000066
R2=1	000065
- FEL-PFL	
730 CONTINUE	000060
IF(.NOT.LEFT) PULAR=148(PELBUF(1.INCOC).AO.1)+1	000067
IF(.NOT.LEFT) GD TO 740	000067
POLAF=1	000067
	00006;
LEFT= FALSE •	000067
740 CONTINUE	00006
C TEST FOR PASS MODE	000067
C	00006
(F1082-05-A1) (d) 10 750	700067
C	100061
PASS MODE CODING (CAN'T END A LINE IN PASS MODELNER AS MUST HAVE	
C SAME POLARITY AS 82)	000066
C	00006
	00006
A0 = 62 \ C0 = 42 \	30006 30006
GO 10 62 J 750 CONTINUE	380000
750 CENT MOE	000000
MAR=JADS(A1=81)	20000
IF (MAB-3) 751,751,799	200065
	00006
C CALCULATE : ENGTH OF VERTICAL AND HERIZONTAL MODES	
C	00006
<u>C</u>	00006:
C. DU HCRIZONTAL FIRST	120000
TEL CONT DUE	00006
751 CONT INUE A1 MAO = A1 - A0	00006
HORIZ= C	00006
1P(41MA0-1L2-63) 93 TO 735	
HO FI Z=COU = (1 + A1 MAO / 64 + 64 + POL AR)	000070
755 CONTINUE	000071
T=VEA, OAP IA)COM=FWEA	00007
HORIZ=HORIZ+CODE(1+TEMP+PCLAR)+3	000076
C CALCULATE VERTICAL LENGTH	00007
	00007
<u>C</u>	00007.
GO TO (760,770,780,780),MAB	00007
STOP 760	000071
2.07.00	00007
760 VERTIC=1	000071
GO_TC 790	00007;
770 VERIIC=3	00007
GO TO 790	00007:
700 VERTIC=0 IF (01-A1.52.3) VERTIC=VERTIC+1	00007 00007
790 CONTINUE	00007
IF (HORIZ- GT-VERTIC) GO TO 835	00007
C C C C C C C C C C C C C C C C C C C	00007
C CODE BY HOWLZONTAL MODE: FIRST DITECT A2	00007.
C	00037.
799 CCATINUE	00007.
IF(I-GT-PSLMAX) GD TD 810	00007.
C CALCULATE POLARITY OF A L	00007.
C CALCULATE POLARITY OF A L	00007.
POLETAR(25L3UF(1.INCOD).AL.L)	00007
800 CONT INUE	00007
PEL=148(PFL 9UF(1 • INCOD) • I • I)	00007
1=1+1	00007
IF (I - L Z - P Z L AX) GD TO 800	00007
IF (1 - L Z - P Z L 4A X) GD TO 800 810 A2 = FEL MAX + 1 GO TO 930	

JNC_ASSIFIED

A2=I	00007. 00007.
A30 COMINIS	00007
CALL CODENCY. POLAT. AO. AZ. COELCT. CODATA)	00007
A0 +A2	00007
40-10-100	00007
C CODE_BY_VERTICAL_NODE	00007
	00007
835 CONTINUE	00007
IF(AISBU ASO.AAO.AAO	00007
840 CALL CODENG(A1-B1+3.0.0.0.0.CDELCT.CCCATA)	00007
48 70 930	
650 CONTINUE	00007
CALL COENG(B1-A1+6,0,0,0,0,CDELCT+CDDATA)	00007
950 CONTINUE AGRAI	00007
	00002
C TEST FOR END OF LINE	00007
	00007
IF (AO. GT. PELMAX) GO TO 210	00007
POL=148(2 & BUF(1.INCO), 40.1)	00007
GO TC 620	00007
210 CONTINUE	00007
C SWITCH CUDE & REFERENCE LINES	0000 <i>T</i>
C SWITCH CODE & REFERENCE LINES	00007
TEMPS I NOTE	
I NAEF= INC CO	00007
INCOD= TEMP	00007
CDEL b= (C) ELCT+ 32-1) /32	00007
CC 300 122 CD5 #	00007
STFBUF(1) =CDBUF(1)	00007
300 CONT INUS	00007
C SAVE LINE LENGTHI DATA DITS + COL)	00007
STAT(INLNCT)=COOATA+13	00007
C CHECK CODES LINE LENGTH	00007
C CHECK CODED LINE LENGTH	00007
Control of Acont A	00007
FILL=C MPM AX - (CDELCT-32) IF(F RL) 400, 400, 250	00007 00007
C CODE LINE TOO SHORT: FILL IT TO CMPMAX	00007
250 CONT INUE	00007
C COELCT=CSELCT+FILL	00007
C ACCUMULATE STATISTICS AND ERROR CORRUPT	
Ċ	30007
400 CONTINUE	00007
C C ERROR CORRUPT	00007 00007
	30007
350 CONTINUE	00007
FREET FERNORS (FRRPNT) - FRROFF - I COM	00007
IF(ERRBIT.LE.0) GO TO 360 IF(ERRBIT.GT.CDELCT-32) GO TO 390	00007 00007
C ERROR IN GANGE OF CODED LINE; CHANGE APPROPRIATE BIT	00007
C BY SEY AND STREET SOME SOME STATE OF THE S	80000
BIY=148(3)FBUF,ERRBIY+32,1) BIY=MCQ(GIY+1,2)	00008 00008
CALL MIZG (GIT-STEBUF-EERBIT+32-1)	20008
ERRCNT=ERRCNT+1	00008
C	60009
C INCREMENT I ROOR LIST POINTER	99998
340 CONTINUE	00000
ERRPNT=ERRPNTFT	00000
IF(ERRPNT .LE.ERRLIM) GO TO 350	องงงร
C ERROR LIST EXHAUSTED	
C ERROR EIST EXPAUSIES	80008 80008
CRAPNT-SCRAPNT-1	
WRITE(4.370) ERRPHT, ERRORS (ERRPHT)	00000
370 FORMAT('DERROR LIST EXHAUSTED AT', 110, 'TH ERRUR;'/	80000
ERFMCD=N.4	80000 80000
man and the second seco	00008

C COMPUTE STATISTICS	000081
390 CONTINUE	380000 380000
T C C EL = T C J EL + C J EL C T - J 2	380 000 38 0 0 00
IF (DIAG) WRITE(6, 160) INLNCT, CODATA	000082
C	000082
IF (.NOY. DI AG) GU TO 460 CDELW= (CDELCT+32-1)/32	000082
MRITEL 6.450) (CORUE(II) .Lal .COELWI	ACCOR:
WRITE(6.450) (STF3UF(I).I=1.CDELW) 450 F0F4AT(6212)	00008
450 CONTINE	- 00000 ÷
FETURN	000083
SOO CONTINUE	00008.
CALL EXIT	00008.
END	.00008.
SUBROUTINE CODENG(MODE, POLAR, A, B, C, COELCT, CODATA)	00008
	-90000 4
COMMCN/3UFF/PELBUF(60.2).CDBUF(240).OTBUF(60.2). * STF9UF(240). STAT(3000)	000084
COMMENZAJFF ZOODE(3.92, 2). CODERD(3.11)	200081
COMMEN/ER AY /ERRORS (2500)	000084
C*************** BEGIN PROGRAM ************************************	<u> 280000</u> .
c state of the sta	000084
CALL # 129 (COBER9(3, MBBE) + COBUF + COELCT+ 1 + CO3ER3 (1 + MBBE) + COBER (1 + MBBE)	-000084
CDELCT =CDELCT +CDDERD(1,MDDE) GD TD (100,200,100,100,100,100,100,100,400,400) .MDDE	000084
	00008:
C MODE 1 2 3 4 5 6 7 8 9 10 11	180000 180000
STCP 129	00008:
C	00008
-C	.00008! .00008
100 CONTINUE	00003
COORTAGE COEATACCOEATACCOEATACCOEATACCOEATACCOEATACCO	180000
RETURN	000081 180000
C HCRIZONTAL 400E(2)	00008
C 200 CONT INVE	00008
CODATA=CODATA+CODERD(1.MCDE)	,00000 ,00009
CALL CODELN(9-4.PDLAR.COELCT.CODATA)	00008
NEWPCL=4J3(POLAR+2,2)+1 CALL CODELN(C-B,NEWPOL,CDELCT,CDDATA)	00008
RETURN	. 00 00 8
C ADD ECLI OR EGLS TO LINE (10.11)	80000
C ASSISTED A ESTA TO THE (10,11)	-00008
800 CCNTINUE	80000
RETURN SNE	00008
SUBROUTINE CHENG(INDEX.COLOR.STATUS.L)	00008
INPLICIT INTEGER(A-Z)	.000003:
C++++++ LABEL20 CO4MON /G32BIT/ ++++++ C	00008 00008:
COMMON /632917/M454(32)vCOMASK(32)vL1017(32)vE2017(32)	-00000
INTEGER 4ASK, COMASK, LIBIT, LZBIT	00008
CCVVCN/3JFF/P1 3JF(60,2),CDBUF(240),OTBUF(60.2),	00008
* STF3UF(240)	00008
COMMONZHJEFZCODE(3.92.2).CODERD(3.11) COMMONZH AYZERRORS (2500)	.8.0008 .8.0000
C+++++++++++++++++++++++++++++++++++++	00003:
COMMON /FILES/TERM.LPFIL.PELFIL.CTFIL.EFFIL	- 0000 a - 00008
C	60000
CARARAMAN AND AND AND AND AND AND AND AND AND A	
C COMCNAIN ARAPEL MAX . VRES . EPHAS E. CMPMAX . ERRMJO . INMAX . K	00009
COMMON /PVAN INLING . OT LING . OT ELW . INELP . COELP . UT ELP . COLL W .	00003
COELCT. INELCT. TCDATA, TCDEL .ERRPNT. = RROFF. = RRLIM.	60000
# INCOD, INREF, DICOD, GIREF, STEBLI	-60003
CM4CN/ICHA9/DD.II.MM.TT.NN.YY	00009
COMMONICISTO/SEARCH DIAG. SYNC. WRITE JZERO LEFT. CHOOL JUNE WHITE	00000
LCGICAL 3 E ARCH, CIAG, SYNC, WRITE .ZERO.LEFT.CHGUL.UNL.WHITE	00009

	000091
C BEGIN DECOJE LOOP: RETRIEVE NEXT CODE WORD LENGTH (L)	000090
SUNITADO 0001 SUNITADO 0001 SOCO=TIENSI SOCOI	000091
CALL GETE CILE MUST VHODE VLDSTOVES	
IF(DIAG) wRITE(6.1003) LENBIT.MODE.LBITS.L 1003 FDFMAT(216.28.16)	000091
GO TO (1040,1200,1205,1190), MCDE	90003
STOP 1040	00009:
IF (LBITS, EQ.CODE(J.INDEX.COLOR)) GO TO 110J	00000
C . THE DESIGNATION OF THE PROPERTY OF THE PRO	00009
C NO MATCH! ASVANCE CODE WORD INDEX VIA DECODE THREAD	00005
C INDEX=CODE(2.INDEX.COLOR) '	00009
IF (INDEX. GE. 93) GO TO 1190	00009
IF(CCDE(1.1NDEX.CDLGR).EQ.LENBIT) GO TO 1040	00009;
C CODE WORD _ONGER; FROM THE TOP	00009
C CODE WORD LUNGER; PROM THE TOP	00009.
C MATCH FOLIA	00009.
C MATCH FCUID	00009.
1100 CENTINUS	00009.
CDELP#CDELP#L	00009.
C NOT AN ECL	00009. 00009.
C	00009.
C TEST FOR MAKE UP OR TERMINATING CODE	00009.
RUNLEN=INCEX-1	00009.
IE (INDEX. 65) RUNLEN=(INDEX-64)+64	2000
IF(FUNLEN.52.0) GD TD 1160 IF(CCLCR.51.1) GD TD 1155	00009. 00009
c	00009
C ADD BLACK RUN TO OUTPUT BUFFER	00005
DO 1150 I=1 •RUNLEN	00005
CALL N 123(C.) OR-1.0 IBUF(1.01COD) .CTELP.1)	_00009
OTELP=CTELP+1	00009
IF (CTELP-1.GT.PELMAX) GD TD 1180	00009.
GC TC 1160	00009
C ADD WHITE RUITE DUTPUT BUFFER (BY DEFAULT)	00005
C ASE WHILE ROLL IS SOLPEL BOFFER (BY DEFAULT)	00009
1155 CONT INUE	00009:
CTELP=CTELP+RUNLEY	00009
IF (STELP-1.ST.PELMAX) GD TD 1180	00009. 00009 .
Č OUTPUT LINE LESS THAN OR EQUAL TO MAX SPECIFIED	00009
	00009
1160 CONTINUE IF (INDEX.LT.65) GO TO 1170	00009
INCEX=3	00009
GO TC 1300	00009
C RUN ACCES FO SUFFUT LINET LENGTH LESS THAN ON CAUGAL TO SELMAX (1)	00009:
c	000091
1170 CONTINUE	00009
CHCUL=•TRUE• STATUS=1	00009
RETURN	
5	00009
C RUN ADDED UNTIL PELMAX EXCEEDED; LINE TCC LCNG (2)	- 00009
1180 CONTINUE	00009
IF(DIAG) WRITE(6.1185) (OTBUF(I.OTCOD).I=1.60)	00009
TIBS FJRVAT(6213) STATUS=2	
RETURN.	00009
C	00009
C NO MATCH FOUND IN CODE TABLE (3)	00009
1190 CONTINUE	0000y
ST ATUS = 3	00009
RE TURN	00009:
(

JNCLASS IF IED

* *** *** *** *** * * * * * * * * * *	000001
	000091
	0000 81 000081
	000091
	000091
Č EOL2 DETECTED (3)	00009
	000094
•	00009
	0000 <i>0</i> 64
	00009
	00009
INFE ICIT INTEGER(A-Z)	00009
COOCONO LABELED COMMUN /G32 BIT/ ++++++	00009
	00010:
CCMMCN /332 81T/MASK(32), CCMASK(32), L181T(32), L281T(32)	00010
	00010
	00010
	00010.
	00010
	00010
Cossociations sections sections	00010
	00010
	00010
COOSSISSION CONTRACTOR VARIABLES CONTRACTOR VARIABLES	00010
	00010
C CC++CN/IV43/PELMAMVVRESVEPHASEVCMPMAXVERFMODVCINMAXVK	00010
	00010
* CCELCT.INELCT.TCDATA.TCDEL.ERRPNT.LRRUFF.ERRLIM.	00010
ERFCYT, INLNCT, CONSEC, CNECKT, LNNGBF, KCNT,	00010
* INCOD, INREF, OTCOD, DTREF, STEBIT	00010
	ocoro
	00010.
LCGICAL SEARCH. DIAG. SY NO. WRITE, ZERO. LEFT, CHCDL. UNE. WHITE	00010
C BEGIN CECODE LOOP: RETRIEVE NEXT CODE WORD LENGTH (L)	00010
	00010
	00010
1002 LENET =CLOSECLO: INDEX)	00010
CALL_GET_E(LENBIT.MODE.LBITS.L)	00010
IF(CIAG) WRITE(6:1003) LENBIT: MODE: LBITS: L	00010
1003 FORMAT(216, 212, 16)	00010
	00010
	00010
IF(LBITS. 23.603ERD(3.INDEX)) GC TC 1100	00010
c	00010
C NO MATCH: ADVANCE CODE MORD INDEX VIA DECODE THREAD	00010
C	00010
INDEX=CODERD(2,INDEX)	00010
15 (102 × 102 × 12) 12 13 140 1 1 10 10 10 10 10 10 10 10 10 10 10 1	00010
IF (CODERD (1.1MDEX).EQ.LENBIT) GO TO 1040	00010
CODE WORD LENGER: FROM THE TOP	00010
c	00010
GC IC 1002	00010
C MATCH FOUND	00010
C MATCH FOUND	00010
1 100 CONTINUE	00010
CDELP= COULP ML	00010
	00013
C NOT AN EOL	00010
	00010
C SING OL AND OL	00010
C FIND 81 AND 82 ,	00010
C FIND B1 AND B2 ,	00010
C FIND B1 AND B2 ,	00010
C FIND 81 AND 82 , C A0=CTELP IF(CTEL9.5).1) A0=0 PDL=CUEDR=1	00010 00010 00010 00013
C FIND 81 AND 82 , A0=CTELP IF(CTELP=5)-1) A0=0 POL=COLOR=1	00010 00010 00010 00010 00010
C FIND B1 AND B2 , C AD=CTELP IF(CTELP.E).1) A0=0	00010 00010 00010 00010 00010 00010
C FIND 81 AND 82 ,	00010 00010 00010 00010 00010 00010 00010
C FIND B1 AND B2 , E	00010 00010 00010 00010 00010 00010 00010 00010
C FIND 81 AND 92 , POLECULOR-1 C DETECT 31 C DETECT 31 C 1=A0+1	00010 00010 00010 00010 00010 00010 00010 00010 00010
C FIND 81 AND 82 , A0=CTELO IF(CTELO.5).1) A0=0 PDU=CDDDR-1 C DETECT 31 C 1=A0+1 IF(1+01-2=L-Max) 60-TO-65 PELMI=0	00010 00010 00010 00010 00010 00010 00010 00010 00010
C FIND 81 AND 92 , A0=CTILO IF(CTELO.5).1) AU=0 PUL=CULUR-I C DETECT 31 C I=A0+1	00010 00010 00010 00010 00010 00010 00010 00010 00010 00010
C FIND 81 AND 82 , A0=CTELO IF(CTELO.5).1) A0=0 PDU=CDDDR-1 C DETECT 31 C 1=A0+1 IF(1+01-2=L-Max) 60-TO-65 PELMI=0	00010 00010 00010 00010 00010 00010 00010 00010 00010

E-13

	PELS[48(3130F(] *31REF) *[*]	301000
	IF(PEL NE PELMI) GO TO 70	00010€
60_	CONTINUS	000106
	PELMIRPEL	000107
	I= I+1	0001 07
	- IF () +LE+ > EL+44) - 60 - 70 - 50	
65	CONT INUE	000107
	81 = 1	000107
	GO YO 92	000107
70	CONT INUE	000107
	IF LEFT NE POLL GO TO 90	000107
	GO TD 60	000107
90	CONTINUE	00010
	814	
	FCL= FEL	0001.08
_		000101
-		002108
5	SELECT SE	00010
C	7-0144	000 101
	<u> 158141</u>	
	IF (I.GT.PS.MAX) © TO 92	0001 08
61	CONTINUE	00010
	- PEL- 1+8(-) TOUF(1+9 TREF) + I + I)	
	IF (PEL -NE -POL) GO TO 92	00010
	I=I+1	000109
	IF(Tale-PELMAX) 50 YD 91	9001 0
52	CONTINUE	000109
	R2±I	00010\$
	GC_TC (100,200,300,400,400,600,600,600),1NDEX	00010
	STCP 100	000109
		
Č F	PASS MCDE	000105
<u>c</u>		000105
100	CONTINUE	00010
	RUNLEN=32 -OTELP	000110
	CHCOL= FALSE	00011.
	GC TC (1155.1145).COLOR	000110
C		00011(
	ORIZONTAL MODE	
Č		000110
_500	CONTINUE	000110
	ENTRY = 3	00011
	CALL ONEENG (ENTRY COLOR STATE . L)	000110
	GD TO (210.1190.1190.1200.1205) STATE	00011
210	CONTINUE	00011
- • -	COLOR=40) (COLOF+2,2)+1	000111
	ENT RY 3	
	CALL ONE NG (ENTRY, COLDR, STATE, L)	00011
	GD TO (220, 1190, 1190, 1200, 1205) , STATE	000111
220	CONTINUE	00011
220	CHCOL= • TRUE •	00011
	GC IC USO	000113
<u></u>		00011
c v	PERTICAL MODE A181=0	00011:
ě		
300	CONTINUE	00011.
200	RUNLEN=31 -OTELP	00011;
	CHCCL=.TRUE.	00011.
	GD TD (11 55.1145).CCLOR	00011
c .		00011
	VERTICAL MADE VRI AIB1=1.2.3	00011.
č	THE PROPERTY OF THE PROPERTY OF	00011
	CONTINUE	
400	E-X30N1+91TC-18=N3JNUR	00011.
	CHCCL= TAUE.	00011.
		00011
C	GG TC (11 55 11 45) + COLUR	00011.
٤		00011.
c \	PRITICAL MODE LEET VILLATED -1 2 7	00011
	/ERTICAL MODE LEFT VLI A 181=1,2,3	00011
C		
-000	CONTINUE	
	RUNLEN=31-JTELP-(INDEX-6)	00011:
	CHCCL= TRUE	00011.
_	GC TC ([155:1145):COLDR	00011.
Ç		00011
	NDC ELACK RU'L TO OUTPUT BUFFER	90011
C		00011
1145	S CONT INUE	00011
	[F (RUNLEA) 1190+1160+1147	
1147	7 CONTINUE	00011
	CO 1150 I =1 :RUNLEN	00011
	CALL MIZ3 (COLOR-1.0TBUF(1.0TCCD), CTELP.1)	00011
	CT ELP= CT ELP+1	00011

IF (OTELP-1.GT.PELMAX) GO TO 1180	00011
60 TC U60	00011
	00011
C ADD WHITE RUN TO OUTPUT BUFFER (BY DEFAULT)	00011 00011
1155 CONTINUE	00011
IF (HUNLEN at 1 .0) GD TO 1190	00011
OTEL PEOTE LP RUNLEN	11000
IF (OTELP-1.GT.PELMAX) GO TO 1180	00011
C RUN ADDED TO OUTPUT LINE: LENGTH LESS THAN ON EQUAL TO PELMAX (1)	00011
CASE CONTINUE CONTINU	00011
STATUS=1	-00011 00011
RETURN	00011
	1 1000
C RUN ACCEC JNT IL PELMAX EXCEEDED; LINE TOO LONG (2)	00011
1180 CONTINUE	00011
IF (DIAG) WRITE(6.1185) (OTBUF(1.0TCOD).1=1.60)	1 1000
1105 FORMAT(6210)	00011
RE TURN	00011
	00011
C NC MATCH FOUND IN CODE TABLE (3)	00011
1190 CONTINUE	00011
STATUS=3	00011
RETURN	00011
C FOLI DETECTED (4)	00011
	00011
1200 CONTINUE	00011
FETURN	00011
C	00011
C EOL2 DETECTED (5)	00011
C 1205 CONTINUE	00011
STATUS=5	00011
FETURN	00011
BL CCK CAT A	00011
c	00011
IMPLICIT INTEGER(4-2) C+************************************	11000
C	00011
COMMON/FILES/TERY.LPFIL.PELFIL.OYFIL.ERFIL	00011
C CCMMCN/BUEF/PEL BUE(60.2).CDBUE(240).CTBUE(60.2).	00011
* STFBUF(240), STAT(3000)	00012
COMMON/HUFF/CODE(3.92.2),CCDERD(3.11)	00012
COMMCY/ERRY/ERRORS(2500) C++++++++++++++++ LABELLEC COMMON VARIABLES ++++++++++++++++++++++++++++++++++++	90018
	00012
CCMMCN/IV AR/PELMAX, VRES, EPHASE, CMPMAX, ERRMOD, LINMAX, K	21000
COMMON/PVAR/INLNNO.OTLNNC.CTELW.INELP.CDELP.UT.LP.CDELW. COELCT.INELCT.TCDATA.TCDEL.ERRPNI.ERRCEF.ERRLIN.	00012
* ERRCNT. INLNCT.CONSEC.ONECNT.LNND3F.KCNT.	00012
* INCOD, INREF, OTCOD, OTREF, STEBIT	00012
COMMENTICHAR/DD:II.WM.TT.NN.YY COMMON ALJGIC/SEARCH.DIAG.SYNC.WRITE.ZERO.LEFT.CHCLL.GNE.WHITE	00012
LOGICAL SEARCH. DIAG. SYNC. WRITE, ZERO. LEFT. CHOOL, ONE, WHITE	00012
	00012
CATA TERM.LPFIL.PELFIL.OTFIL.ERFIL/5.6.1.2.3/	00012
DATA PEL4AX. VRES. EPHASE. CMPMAX.ERRMOD.LINMAX/1728.2.3.96. T. 3000/	00012
CATA K/2/	21000
C	21000
DATA_CODE(1, 1,1),CODE(2, 1,1),CODE(3, 1,1)/ 8, 70,20035/	00013
DATA CODE(1, 2.1).CODE(2, 2.1).CODE(3, 2.1)/ 6, 90.20007/ DATA CODE(1, 3.1).CODE(2, 3.1).CODE(3, 3.1)/ 4, 4.20007/	00013
DATA CODE(1. 3.1).CODE(2. 3.1).CCDE(3. 3.1)/ 4. 4.20007/ DATA CODE(1. 4.1).CODE(2. 4.1).CCDE(3. 4.1)/ 4. 5.20008/	00013 00013
DATA CODE(1. 5.1).CODE(2. 5.1).CODE(3. 5.1)/ 4. 6.Z000B/	00013
DATA CODE(1, 6.1), CODE(2, 6.1), CCDE(3, 6.1)/ 4. 7, Z000C/	00013
DATA C33E(1: 7:1):C30E(2: 7:1):CC0E(3: 7:1)/ 4: 3:2000E/ DATA C33E(1: 8:1):C30E(2: 8:1):C00E(3: 8:1)/ 4: 9:2000F/	00013
DATA CODE(1, 9.1).CODE(2, 9.1).CODE(3, 9.1)/ 5, 10.Z0013/	00013
DATA CDJE(1, 10,1),CJDE(2, 10,1),CCDE(3, 10,1)/ 5, 11,20014/	00013
DATA CODE(1, 11,1),CODE(2, 11,1),CCDE(3, 11,1)/ 5, 12,Z0007/	00013

DATA CODEC 1.	12:11:270512:	17.11.10061	1. 12:117 S.	65.7000H7	00013
DATA CODE(1.	13.1).CODE(2.	13.1).CODE(3, 13,1)/ 6,	14.20008/	00013
DATA COVERY	LALLACODE (2.	TA-TI-CODE!	3 - 14 - 1/4	15.20003/	-00013
DATA CODE(1.					00013 [']
-DATA COOE(14-	17,1),CODE(2,	-17v1) vc coc (3v 17v1)/ 6v	1 U V Z O O E A /	-00013
DATA CODE(1.	18.1).CODE(2.	18.1).CDDE(3. 18.1)/ 6.		00013
DATA CODE(1.	20.11.CUDE(2.	20.1).CDDF(3. 20.11/ 7.	20.20027/	00013
DATA COJE(1. a	21 •1	21.1),CCDE(3. 21.11/ 7.	22.Z0008/	00014
DATA CODELLA	22.1).CODE(2.	22.1).CDDE(3. 22.11/.7.	24.70017/	00014
DATA CODE(1.	24.1).CDDE(2.	24.1) .CDDE (3. 24.1)/ 7.	25.Z0004/	00014
DATA COJECTA	25 v1) v CODE(2v	-25+ 1)+ CB DE (3+ 25+11 / 7+	- 10300 TAGE	00014
DATA CODE(1.	26.1).CODE(2, 27.1).CODE(2.	26,1),CCDE(3, 26,11/ 7,	27 • Z002B/	00014
DATA COSETT.					00014
DATA CODE(1.	29.1).CODE(2.	29,1),CGDE(3. 29.11/ 7.	68.Z0018/	00014
DATA CODE(1.					00014
	32.1).CODE(2.				00014
- 0474-63)2(1+-	33 +t) +C30E+2 +	33-11-C00E1	3+ 33+13/ 3+	-34 v 200 t D/	- 00014
DATA CODE(1.					00014
DATA CODECT.	36.11.CJDE(2.	36.17.CODE(3. 35.11/ 8.	37.200147	00014
	37,1),CODE(2,				00014
DATA CODE(1.	39.11.CDDE(2.				00014
DATA CODE(1.	40 .1) . CUCE (2 .	40.1),CCDE(3. 40.11/ B.	41.20028/	00014
DATA CODE(1-	*1+1++CODE(2+ 42+1).CODE(2.				
	43.1) .CODE(2.				00014
DATA CODECT.			3. 44.177 5.		00014
DATA CODELLA	45.1).CODE(2.				00014
DATA CODE(1.					00014
DATA CODE(1.	48.1).CODE(2.	48,1),CODE(3, 48,1)/ 8,	49.Z000A/	00014
	50.1) .CJDE(2.				00014
DATA CODE(1.	51.1).CODE(2.	51,1),CODE(3. 51.1)/ 8.	52.20053/	00014
	52.1).CDDE(2.			53. Z0054/	00014
DATA CODE(1.	54-11.CDDE(2.	54.11.CODEL	3. 54.11/ B.	55.70024/	00014
	55.1).CODE(2.				00014
- DATA - COJE(1)	56,1),CJDE(2, 57,1),CJDE(2,	-57-11-CCD2(31 301117 31 31 571117 61	-58+20059/	00014 00014
DATA CODE(1.	58.1).CODE(2.	58.1).CODE(3. 58.11/ 8.	59,Z005A/	00014
	59.1), CODE (2.				00014
DATA CODE(1.	61.1).CJDE(2.	61.1). CB CE(3. 61.11/ 6.	62.Z0048/	00014
DATA CODE(1.	62.1).CDDE(2. 63.1).CDDE(2.		3. 62.11/ 8. 3. 63.1)/ 8.		00014
	64,1),CDDE(2.				00014
-0ATA-033E(-1					- 00014
	66,1),CDDE(2, 67,1),CDDE(2,				00014 00014
DATA CODECI.	68,11,CODE(2,	68,1),CODE(3, 68, 1) / 7,	30,200377	00014
DATA CODE(1.	69.1).CDDE(2.70.1).CDDE(2.	70.11.CODE(3. 69.11/ 6. 3. 70.11/ 8.	71 20036/	00014
DATA CODE(1.	71 ,1) , CODE (2.	71.1), CODE (3. 71.11/ 8.	72.20064/	00014
DATA CODE(1.	72,1),CODE(2,				00014 00014
	74.1) .CODE(2.				00014
	75.1), CODE(2.	75.11.CODE(3. 75.1)/ 9. 3. 76.1)/ 9.	76.200CC/ 77.200CD/	00014
DATA CODECL.	76.1),CODE(2.	77.1).CDDE(3. 77.11/ 9.	78.20002/	00014
DAIA_CODE(1	ZB.LJ.CODE(2.	78.11.CODE(378.11/ 9.	79.20003/	0001.4
	79.1).CDDE(2.80.1).CDDE(2.				00014
	01 v1) vC00E12 v	-01+1++CO0E(3+ 41+11 5+	-32 / 200 D6 /	
DATA CODE(1,	82, 1), CODE (2, 33,1), CODE (2,	82.1),CDDE(3. 83.11/ 9.	83 •Z00D7/	00014
DATA CODECT.	34,17,CUDE(2.	84.17.CODE (3. 34.177 5.	55,200097	00314
1) 2 GCD ATAC	35.1).CJDE(2.	85.1). CODE	3. 85.1) / 9.	86.Z000A/	00014
DATA CODELL	37.1).CDDE(2.				00014
DATA CODE(1.	88.1) . CODE (2.	88 .1) . CCDE (3. 35.1)/ 9.	89.Z0099/	00014
DATA CODELLA	99,11,C35E(2, 90,1),C30E(2,	90411 CODE	30 49 11 1 6.	13.70018/	
DATA CODECI.	91.1).CDDE(2. 92.1).CDDE(2.	91.11,CODE	3. 91.11/ 9.	32.2009B/	00014
DATA CODE(II	92.11.CUDE(2.	92.1).CDDE(3. 92.11713.	93.200037	00014
DATA CODE(1.	1.2).CODE(2.		31 116//100	00,2000//	00014

DAYA COSECI.	2.21.CDE(2.	2.21.CODE (3.	2.217 3.	6.20002/	000147
DATA CODE(1.	3.2).CODE(2.	3, 2), CDDE(3,	J. 2)/ 2.	4.Z0003/	000147
DATA COJELLA	A.21 .CODE(2.	4.21.CCDE(3.	_4.21/ 2.	5.Z0002/	00014
DATA CODE(1.	5.2).CODE(2.	5.2).CODE(3.	5,2)/3.	2.20003/	000147
DATA CODE(1.	6,2),CODE(2,	6,2).CODE(3.	6,2}/ 4,	7, Z0003/	00014
DATA COOCI IV	7v 2) vCODE(2v	7v2)v000E(3v	-7+2}/ ++		- 00014i
DATA CODE(1.	9.2),CODE(2,	8.2), CDDE(3.	8.21/ 5.	9.Z0003/	00014
DATA CODE(1.	9.2) .CODE(2.	9.2) .CODE(3.	9.21/6.	10.Z0005/	00014
TITSCO ATAU	10.21.0006(2.	10,21,CODE(J.	10.2)/ 6.	11,20004/	000148
DATA CODE(1.	11.2) .C JDE (2.		11,21/ 7.	12.Z0004/	000148
DATA COSELLA	12.2) CODE (2.	12.2) .CDDE(3.		13.20005/	000145
DATA CODELL.	13.2), CODE (2.		13.21/ 7.	14.Z0007/	00014
DATA CODEC L.	14.2).CODE(2.		14.21/ 8.	15.Z0004/	00014
	-15v2) v CODE (2v			16v20007/	
DATA CODECL.	16.2) .CJDE(2.	16.2).CCDE(3.		17.Z0018/	00014
DATA CODE(1.	17.2).CODE(2.		17,21/10.	18.20017/	000148
DATA CODE (I.	IBIZTICOCETZ.	13.21.CODE(3.	18.21/10.	19,20018/	00014
DATA COSE(I.	19.2) .CODE (2.	19,2) .CODE (3.	19.2)/10.	1.20008/	00014
DATA COJECLA		20.2). CDDE(3.	20.21/11.		00014
DATA CODE(1.	21.2),CJDE(2.	21.2).CODE (3.		22.Z0068/	00014
DATA CODELL.	22.2), CDDE(2.	22,2),CODE(3.	22.21/11.	23.Z006C/	00014
DATA COSELLY				24 - Z0037/	- 00014
DATA CODEC 1.	24.2).CODE(2.	24.2),CODE(3.		25.Z0028/	00014
DATA CODELL.		25.2). CODE(3.		26.20017/	00014
DATA CODE(1.	26.21.CUDE(2.	26.21,CODE (3.	25.21/11,	27.20018/	00014
DATA CODE(1.	27.2).CODE(2.	27, 2), CDDE(3,		28.Z00CA/	00014
DATA CODECTA	28.21.CDDE(2.	28.21.CDDE(3.			00015
DATA COSE(1.	29.2).CODE(2.	29.2).CODE(3.		30 .Z 00CC/	00015
DATA CODELL.	30.2).CDDE(2.	30 .2).CODE(3.	30.21/12.	31.Z00CD/	00015
DATA CODELLA	- 31 + 21 + C 30 E (2 +				
DATA CODE(1.	32,2), COCE(2,	32.2).CGCE(3.		35.20069/	00015
DATA CODE(I.	33.2).CODE(2.	33.2).CODE(3.	33,21/12	34.2006A/	00015
DATA CODE(1.	34.27.CJDE(2.	34,2),0008(3,	34.21/12.	35.Z006B/	00015
DATA CODE(1.	35.2).CODE(2.		35,2)/12.	35.20008/ 35.20002/	00015
DATA CODELLA	36.2).CJDE(2.			37.Z00D3/	00015
		37.2). COCE(3.			
DATA CODE(1.	37.2).CUDE(2.			38.200D4/	00015
DATA CODE(1.	38.2).CODE(2.	38.2).CODE(3.		39.Z00D5/	00015
DATA CODE(1-v				-40+20056/	00015
DATA CODE(1.	40,2),CJDE(2,			41 .Z00D7/	00015
DATA CODE(1.	41,2),CODE(2,	41,2),CODE(3,	41.2)/12.	42.Z006C/	00015
DATA CODE(I.	42.2).CJCE(2.		42.21/12.	43, Z006D/	00015
DATA CODE(1.	43,2),CODE(2,	43,2),CDDE(3,	43.21/12.	44 . ZOODA/	00015
DATA CODE(1.		44.2). CODE(3.		<u>45.70038/</u> 42.20054/	00015
	45.2) .CDDE(2.	45,2),CODE(3,			00015
DATA CODE(1.	46,2),CDDE(2,		46, 2) / 12,	47.20055/	00015
		-47,2),600E(3,		+9+20056/	00015
DATA CODE(1.	48.2).CODE(2.	48.2).CODE(3.		49,20057/	00015.
DATA CODELL.	49,2),CODE(2,		49.21/12.	50 · Z 0064/	00015
DATA CODE(I.	50.2).CODE(2.		50.2)/12,	51 . Z0065/	00015.
DATA CODE(1.	51.2).CODE(2,	51.2).CODE(3.	51.21/12.	52.Z0052/	00015
DATA CODEL 1+		_52.2).CCDE(3.			00015
DATA CODE(1.	53.2).CDDE(2.	53.2).CODE(3.	53.21/12.	54,20024/	00015
DATA CODE(1.	54.2).CODE(2.		54.2)/12.	55.Z0037/	00015
DATA CODECT	-55,2),000E(2,			- jo v 20 038/	00015
DATA CODE(1.	56.2).CJDE(2.	56,2),CODE(3,	56.21/12.	57.Z0027/	00015
DATA CODE(1.		57.2).CODE(3.		33.20028/	00015
DATA CODE(1,	53,2),CODE(2,	53,2),0002(3,	58.2)/12.	59.Z0058/	00015
DATA CODE(1.	59.2) .CODE(2.	59.2).CCDE(3.	24161/161	6J.Z0059/	00015
DATA COST	- 304 CJ 4 CJ 3 C (2 4	DU-ZI-CUDEL 3-	- 0 0.421/12.	01 • Z 0 0 2 BZ	00015
		61.2). CODE(3.			00015
		62,2),CODE(3,			00015
DATA CONSTITUT	64 21 6205(2)	-63+2)+CDDE(3+	54-1111	55 70067	- 00015
		64.2).CCDE(3.			00015
		65.2).CCDE(3.			00015
DATA CODECT		66.2).0005(3.			00015
DATA CODELLA	49.21.0305(2)	67.2),CODE(3, 68.2),CODE(3.	62.21/121	64.700CY/	00315
		69.2) .CODE (3.			00015
DATA CODE(1,	70.21.0308(2.	70.2),CDDE(3.	10121/121	71.20034/	00015
		72.2).CODE(3.			00015
DATA CODE(1)	73.31 (305(2)			74.20062/	
DATA CODE(1.	73.2),CJDE(2,			75.2003A	00015
DATA CONT.	75.21.020512	74.2).CCDE(3. 75.2).CODE(3.	75, 21 / 12		00015
		76.2).CCDE(3.		76•Z004B/	00015
		77,2),CODE(3,		70.2004D/	00015
2414 C22411	78-21-0300(2)	78.2).CODE(3.	74.21/13.	79.20072/	
		79,2),CCDE(3,			00015
					00015
		80,2),CODE(3,			00015 00015
53+2 23<3\11	32.31.00000(4)	81,2),CODE(3, 82,2),CODE(3,	33.317131	61.710767	00015
DATA CODE(1)	9212110000121	83.2).CCDE(3.	72.51/13	342 700777	00015
C37E([)	03127 10302(2)	33 12 / 1 CODE (3)		2 T T L V V I //	20013

-		
	DATA CIDECI, 94.21.CDDEC2. 84.21.CDDEC3. 84.21713. 85.200527	00015
	ATA CODE(1. 85.2).CODE(2. 85.2).CODE(3. 85.2)/13. 80.20053/	00015
	DATA CDE 1 - 86-21 - CDE (2 - 86-21 - CDE (3 - 86-2) / 13 - 37 - 20054/	00015
	DATA CUDE(1. 87.2). CODE(2. 87.2). CODE(3. 87.2)/13. 68.20055/	00015
	VACOUSTIES - 61/17 PRO 19 - 19 - 19 - 19 - 19 - 19 - 19 - 19	00015
	0ATA C33E(11 0912)1C00E(21 0912)1C00E(31 0912)/147 9012050/	00015
	DATA COJE(1. 90.2).CODE(2. 90.2).CODE(3. 90.2)/13. 91.Z0064/	00015
	DATA CD)E(1, 91.2).CDDE(2, 91.2).CODE(3, 91.2)/13.)3.Z0065/	00015
	DATA CUSE(1, 92.2), CSDE(2, 92.2), CDDE(3, 92.2)/13, 93.20003/	00012
	DATA COJERU(1:1).CODERD(2:1).CDDERD(3:1)/ *.5.21/	00015
	DATA CDJERD(1-2).CDJERD(2-2).CDDERD(3-2)/ 3-4-23/	00015
	DATA CODERO(1.3), CODERD(2.3), CODERD(3.3)/ 1.2.21/	00015
	DATA CODERO(1.4).CODERO(2.4).CODERO(3.4)/ 3.7.Z1/	00015
		-00015 -
	OATA CDJERD(1.6).CODERD(2.6).CODERD(3.6)/ 6.8.21/	00015
	DATA CODERS(1,7), CODERD(2,7), CODERD(3,7)/ 3,1,22/	00015
	DATA CODERD(1.8).CODERD(2.8).CODERD(3.8)/ 6.9.22/	00013
	DATA COJERO(1.9).COJERD(2.9).CODERD(3.9)/ 7.10.21/	00012
	DATA CODERD(1.10).CODERD(2.10).CODERD(3.10)/ 13.11.23/	00015
	DATA CODERD(1:11):CODERD(2:11):CODERD(3:11)/ 13:12:42/	00015
C		00015
	E N D	- 00015
	SUBROUTINE CODELN(LENGTH, POLAR, CDELCT, CDDATA)	00015
C		00015
	IMPLICIT INTEGER(A-Z)	00015
	CDMMON/9JFF/PEL8JF(60.2).CDBUF(240).OTBUF(60.2).	00015
	\$ STERUF(2401, STAT(3000)	00015
	COMMON/HUFF/CODE(3,92,2),CODERD(3,11)	00015
	CCMM CN/ER AY / ERRORS (2500)	00015
€		00015
Ć*	*****************************	
		00015
<u>c</u> _	INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH	00015
č		00015
	MC CDE=0	00015
	ML EN G= 0	00015
C		00015
ě	CHECK INPUTS	00015
č		00015
-	IF(PCLAR.LT.1.0R.POLAR.GT.2) CALL EXIT	00015
	IF (LENGTH T.O. UR. LENGTH GT. 1728) CALL EXIT	00013
C		00015
	IF(LENGTH-LE-63) GO TO 10	00016
~		00016
Ç	CALCULATE MAKE UP CODE INDEX. CODE. LENGTH	00016
ě	AND WRITE TO CODE LINE	00010
č		00016
•	INDE X=LENG TH/64+64	00016
	PCCSE=CSSE(3, INDEX, POL AR)	8 1000
	MLENG=COJE(I, INDEX, POLAR)	00016
	CALL MIZHINCODE COBUE COELC T+ LAMLE NG)	00016
	CDELCT =CJELCT + VLENG	00016
	CODA TA =CODA TA+ 4LENG	00016
-		-0001-6
č	CALCULATE TERMINATING CODE INDEX. CODE. LENGTH	00016
č	AND ADD TO CODE LINE	00016
듣		00016
-	10 CONTINUE	00016
	INDEXECTION (LENGTHE 64) + LENGTHE	00016
	TC CDE=CODE(3.INDEX.POLAR)	00016
	TL ENG= COO E(1, INDEX, POLAR)	00016
	CALL WIRD (TCS 92-CONUTY COELCT +1-TLENG)	-00016
	CDELCT COELCT+TLENG	00016
	CORATA -CARA ATI ENG	00016
<u> </u>	COURTACUSTATICE	61000
-	RE TURN	00016
	ENC	
	SUBRCUTI NE ERRYES(PELBUF, OTBUF, PEL MAX, VRES, ERRCHT)	00016
C		00016
		-0001÷
	REAL ESF	00016
C*	***** LARE 2) COMMON /G3281T/ ******	00016
<u> </u>	·	5 10 00 -
-	COMMON /j3291T/MASK(32).COMASK(32).Ll8IT(32).L28IT(32)	00016
		00016
C		00016
Č*	******************* FILE DEFINITIONS **************	00016
<u>-ě</u>		00010
•	COMMON/FILES/TERM.LPFIL.PELFIL.CTFIL.ERFIL	00016
_C		
	DIMENSION PELBUF(50) CYBUF(60)	31000
	CDMMON/_JGIC/SEATCH.DIAG	00016

JNCLASS IF IED

LUGICAL SEARCH, DYAG	00016
C. C	00016
C	00016
REWIND PELFIL	00016
ERROR=0	99016
OTELW=(P3L4AX+32-1)/32	00016
CTUNCT=0	00016
C READ AN ERROR FREE LINE	00016
	00016
100 CONTINUE	00016
IF (MCD(I.NLNNO-1.VRES).NE.0) GD TO 100	00016
	00016
C READ AN ERROR-CORRUSTED LINE	9 1000
C	00016
READ(2.EX)=500, ERR=800) CTLNNO.OTELCT.OTBUF	91000
OTLNCT=OTLNCT+1	00016
C C	99916 00016
C COUNT DIFFERENCES BETWEEN TRANSMITTED AND RECEIVED LINES	00016
	9 1000
DO 450 I=1.0TELW IF(OTBUE(I).E).PFLBUE(I)) GO TO 450	00016 00016
IF(.NOT.JIAG) GO TO 420	00016
WRITE(6.:10) INLNYD.OTLNNO.I.PELBUF(I).OTBUF(I)	91006
420 CONTINUE	
CO 440 J=1, 32	00016 00016
IF (148() TBUF(1).J.1). NE. 148(PELBUF(1).J.1)) ERRCK=ERROR+1	00016
440 CONTINUE	00016
IF (OTLNN)-INLNN) 200,100,580	00016
C	00016
C COUNT DIFFERENCES BETWEEN GOOD AND ALL WHITE LINE	00016
C COONT DIFFERENCES SETWEEN GOOD AND ALL WHITE LINE	00016 00016
50 0 CONTINUE	00016
CO 550 I=1.0T5LW IF (PELBUF(11.50.0) GO IC 550	01000 01000
IF(-NOIT-DEED CT CD (DAI C-TCM-)I	00016
WRITE(6.410) INLNNO,ÖTLNNO,I,PELBUF(I),OTBJF(I)	00016
520 CSHTINUE CC 540 J=1.32	90016 00016
IF(I4B(PIL3UF(I).J.1).NE.O) ERROR=ERROR+1	00016
540 CONTINUE	91000
550 CONTINUE	00016
580 FEAC (1.EN D=600, ERR=800) INLNNO, INEL CT. PELBUF	00016
IF(MCD(INLNNO-1.VRES).NE.O) GO TC 580	00016
GO TO 300	00016
	00016
C CALCULATE LARDA SENSITIVITY FACTOR	00016
C 600 CONTINUE	01000
ESF=0.	00016
IF(ERRCNT.LS.0) GD TO 650	00017
### ### ##############################	90017 00017
C	00017
WRITE(6.700) ERRUR, FRRCHT, ESF, OTLNCT	00017
700 FORMAT('ONUMBER OF INCORRECT PELS = '. 110/	00017
* 'JERROR SENSITIVITY FACTOR = 1 . F12 . 4/.	00017
* 'OTOTAL NUMBER OF OUTPUT LINE'S PROCESSED = '.18)	00017
RE TURN	90017 90017
800 CONTINUE	00017
51CP 800	00017
END SURROUTINE STATS (LENGT HAINLINGT ADIAG)	00017
IMPLICIT INTEGER (A-Z)	00017
C	00017
REAL STI(2, 5), SUM, SUM SQ	00017
LOGICAL DIAJ	00017
Cassassassassassassassassassassassassass	00017
<u> </u>	00017

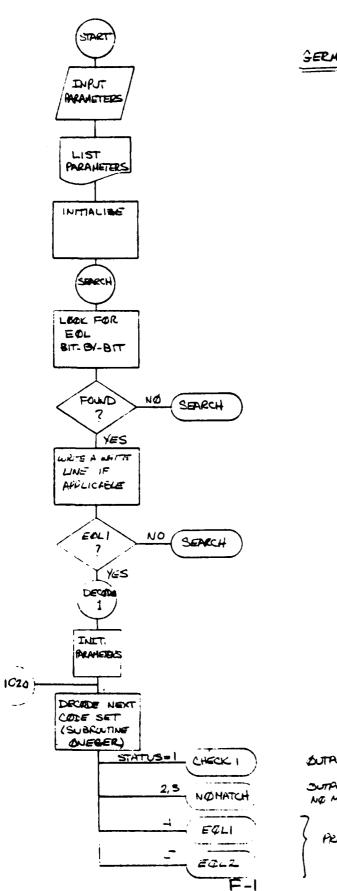
# CODED LINE // # LENGTH 0 MS 5 MS 10 MS 20 MS # STATISTICS: // # MINITUM * 10%,5(19) //) WRITE(6,410)(ITT(2,1),1=1.5) 410 FORMAT(# MAXIMUM * [OX.5(18) //) WRITE(6,420)(STT(1,1),1=1.5) A20 FORMAT(# SA MPLE MEAN*,9X.5(F8.2) //)	100 100		ELFILTOTF	IC + CKL I	L			000
OD 3C0 I=1.5 ITY(1, I)=10000 ITY(2, I)=0 SUM=0. SUMSCAD. DD 100 J=1.INLNCT FIND FILLTD LING LENGTH LEN=MAXO(L_INGTH(J),MTT(I)) IF(DIAG) WITYE(6,50) LEN 50 FORMAT(I8) FIND MINIMUM LINE LENGTH ITT(1, I)=MINO(LEN,ITT(1,I)) FIND SUM OF LENGTHS SUMSOB SUM SI+(FLDAT(LEN)) **2 100 CONTINUE FIND SAMPLE MEAN AND STANDARD DEVIATION STT(1, I)=SUM/FLDAT(INLNCT) STT(2, I)=S)RT((SUMSQ-(SUM**2)/FLOAT(INLNCT))/FLDAT(INLNCT) **O SUMSTINUE WRITE(6,40)(ITT(1,1).I=1.5) **O FORMAT(**O MINIMUM TRANSMISSIJN TIME (48) **O STATISTICS:// **O MS 5 MS 10 MS 20 MS **O MINIMUM TRANSMISSIJN TIME (48) **O STATISTICS:// **O MS 5 MS 10 MS 20 MS **O MINIMUM TRANSMISSIJN TIME (48) **O MS 5 MS 10 MS 20 MS **O MINIMUM TRANSMISSIJN TIME (48) **O MS 5 MS 10 MS 20 MS **O MINIMUM TRANSMISSIJN TIME (48) **O MS 5 MS 10 MS 20 MS **O MINIMUM TRANSMISSIJN TIME (48) **O MINIMUM TRANSMISSIJN TIME (48) **O MINIMUM TRANSMISSIJN TIME (48) **O MS 5 MS 10 MS 20 MS **O MINIMUM TRANSMISSIJN TIME (48) **O MINIMUM TRANSMISSIJN TIME (48) **O MS 5 MS 10 MS 20 MS **O MINIMUM TRANSMISSIJN TIME (48) **O MS 5 MS 10 MS 20 MS **O MINIMUM TRANSMISSIJN TIME (48) **O MINIMUM TRANSMISSIJN TIME (48) **O MS 5 MS 10 MS 20 MS **O MS 5 MS 10 MS 20 MS **O MINIMUM TRANSMISSIJN TIME (48) **O MS 5 MS 10 MS 20 MS **O MS 10 MS 20	000 000							000
ITT(1,1)= 10000 ITT(2,1)= 0 SUM=0. SUMSCAD. DD 100 J=1.INLNCT FIND FILED LING LENGTH LEN=MAXO(L_SIGTH(J), MTT(I)) IF(DIAG) MITTE(6,50) LEN 50 FORMAT(IB) FIND MINIMUM LINE LENGTH ITT(1,1)= MINO(LENVITT(1,1)) FIND SUM OF LENGTHS SUMSCAUS SUMSOM (FLOAT(LEN))**2 100 CONTINUE FIND SAMPLE MEAN AND STANDARD DEVIATION STT(1,1)= SUM/FLOAT(INLNCT) STT(2,1)= S)RT((SUMSQ-(SUM**2)/FLOAT(INLNCT))/FLOAT(INLNCT) WRITE(6,400)(ITT(1,1),1=1,5) #*O FORMAT(**O MINIMUM TRANSMISSION TIME (48) **O STATUSTICS:*// WRITE(6,400)(ITT(2,1),1=1,5) #*O MS S MS 10 MS 20 MS **O MINIMUM TRANSMISSION TIME (48) **O MS S MS 10 MS 20 MS **O MINIMUM TRANSMISSION TIME (48) **O MS S MS 10 MS 20 MS **O MINIMUM TRANSMISSION TIME (48) **O MINIMUM TRANSMISSION TIME (48) **O MINIMUM TRANSMISSION TIME (48) **O MS S MS 10 MS 20 MS **O MINIMUM TRANSMISSION TIME (48) **O MS S MS 10 MS 20 MS **O MINIMUM TRANSMISSION TIME (48) **O MINIMUM TRANSMISSION TIME (48) **O MS S MS 10 MS 20 MS **MINIMUM TRANSMISSION TIME (48) **O MS S MS 10 MS 20 MS **MINIMUM TRANSMISSION TIME (48) **O MS S MS 10 MS 20 MS **MINIMUM TRANSMISSION TIME (48) **O MS S MS 10 MS 20 MS **MINIMUM TRANSMISSION TIME (48) **MINIMUM TRANSMISSION TIME	000 000 000 000 000 000 000 000 000 00		PROGRAM**	*****	*****	*****	******	
SUMSCO. SUMSCO. DO 100 J=1,INLNCT FIND FILLED LING LENGTH LEN=MAXO(L:NGTH(J),MTT(I)) IF(DIAG) WRITE(8.50) LEN 50 FDRMAT(I8) FIND MINIMUM LINE LENGTH ITT(1*)=MINO(L:N*,ITT(1*)) FIND MAXIMU4 LINE LENGTH ITT(2.1)=MAXO(LEN*,ITT(2.1)) FIND SUM OF LENGTHS SUMSQ=SUMSL*,AT(LEN) SUMSQ=SUMSL*,AT(LEN) STT(1.1)=SUM/FLOAT(LEN))**2 FIND SAMPLE MEAN AND STANDARD DEVIATION STT(1.1)=SUM/FLOAT(INLNCT) STT(2.1)=S)RT((SUMSQ-(SUM**2)/FLOAT(INLNCT))/FLOAT(INLNCT) WRITE(6.400)(ITT(1.1).I=1.5) 400 FDRMAT(WRITE(6.400)(ITT(2.1).I=1.5) WRITE(6.400)(ITT(2.1).I=1.5) WRITE(6.400)(ITT(2.1).I=1.5) WRITE(6.400)(ITT(2.1).I=1.5) WRITE(6.400)(STT(1.1).I=1.5) WRITE(6.400)(STT(1.1).I=1.5) AD FDRMAT(WRITE(6.400)(STT(1.1).I=1.5)	000 000 000 000 000 000 000 000 000 00							
SIMSCED. DO 100 J=1,INLNCT FIND FILED LING LENGTH LENEMAXO(LENGTH(J),MTT(I)) IF(DIAG) WITE(6.50) LEN 50 FORMAT(I8) FIND MINIMUM LINE LENGTH ITT(1*1)=*MINO(LEN*ITT(1*1)) FIND MAXIMUM LINE LENGTH ITT(2.1)=*MAXO(LEN*ITT(2.1)) FIND SUM OF LENGTH SUMSCESUM-SJ*(LEN) SUMSCESUM-SJ*(LEN) SUMSCESUM-SJ*(FLDAT(LEN))**2 100 CONTINUE FIND SAMPLE MEAN AND STANCARD DEVIATION STT(1.1)=*SUM/FLDAT(INLNCT) STT(2.1)=*SIRT((SUMSQ-(SUM**2)/FLDAT(INLNCT))/FLDAT(INLNCT) WRITE(6.4 0.3) (ITT(1.1).I=1.5) #** LENGTH O MS 5 MS 10 MS 20 MS #** LENGTH O MS 5 MS 10 MS 20 MS #** STATISTICS:*// *** STATISTICS:*// *** MINIMUM TRANSMISSION TIME (46 #** LENGTH O MS 5 MS 10 MS 20 MS #** LENGTH O MS 20 MS 20 MS #** LENGTH O MS 20 MS 20 MS 20 MS #** LENGTH O MS 20 MS 20 MS #** LENGTH O MS 20 MS 20 MS 20 MS #** LENGTH O MS 20 MS 2	000 000 000 000 000 000 000 000 000 00							000
### PILLES LINE LENGTH LEN=MAXO(LENGTH(J), MTT(I)) IF(DIAG) WITE(6.50) LEN 50 FORMAT(I8) FIND MINIMUM LINE LENGTH ITT(1.1)=MINO(LEN.ITT(1.1)) FIND MAXIMUM LINE LENGTH ITT(2.1)=MAXO(LEN.ITT(2.1)) FIND SUM OF LENGTHS SUM=SUMF(L)AT(LEN) SUMSOS SUM SJ*(FLDAT(LEN))**2 100 CONTINUE FIND SAMPLE MEAN AND STANDARD DEVIATION STT(1.1)=SUM/FLDAT(INLNCT) STT(2.1)=SIRT((SUMSQ-(SUM**2)/FLDAT(INLNCT))/FLDAT(INLNCT) WRITE(6.403)(ITT(1.1).I=1.5) 400 FORMAT(*' O	000 000 000 000 000 000 000 000 000 00	SUNS CEO.	······································					مَمَمَ
LEN=MAXO(Lingth(J), MTT(I)) IF(DIAG) WRITE(6.50) LEN 50 FORMAT(I8) FIND MINIMUM LINE LENGTH ITT(1.1)= MINO(LEN.ITT(1.1)) FIND MAXIMUM LINE LENGTH ITT(2.1)= MAXO(LEN.ITT(2.1)) FIND SUM OF LENGTHS SUM=SUM-FLAT(LEN) SUMSQ=SUM-SLAT(LEN) SUMSQ=SUM-SLAT(LEN))**2 100 CONTINUE FIND SAMPLE MEAN AND STANCARD DEVIATION STT(1.1)= SUM/FLOAT(INLNCT) STT(2.1)= SIRT((SUMSQ-(SUM**2)/FLOAT(INLNCT))/FLUAT(INLNCT) WRITE(6.40)(ITT(1.1).I=1.5) 400 FORMAT(*' CODED LINE'/ *' LENGTH *' STATISTICS:'// *WRITE(6.40)(ITT(2.1).I=1.5) 410 FORMAT(*V MAXIMUM': [OX.5(I8)//) WRITE(6.40)(STT(1.1).I=1.5) 420 FORMAT(*V MAXIMUM': [OX.5(I8)//) WRITE(6.420)(STT(1.1).I=1.5) A20 FORMAT(*' SAMPLE MEAN'.9X.5(F8.2)//)	000 000 000 000 000 000 000 000 000 00							000
IF (DIAG) WRITE(6.50) LEN SO FORMAT(18) FIND MINIMUM LINE LENGTH ITT(1*I)= MINO(LEN*ITT(1*I)) FIND MAXIMUM LINE LENGTH ITT(2*I)= MAXO(LEN*ITT(2*I)) FIND SUM OF LENGTHS SUM= SUM*FL)AT(LEN) SUMSQ= SUM SJ*(FLDAT(LEN)) **2 100 CONTINUE FIND SAMPLE MEAN AND STANDARD DEVIATION STT(1*I)= SUM/FLDAT(INLNCT) STT(2*I)= SIRT((SUMSQ-(SUM**2)/FLOAT(INLNCT))/FLDAT(INLNCT) STT(2*I)= SIRT((SUMSQ-(SUM**2)/FLOAT(INLNCT))/FLDAT(INLNCT) STT(2*I)= SIRT((SUMSQ-(SUM**2)/FLOAT(INLNCT))/FLDAT(INLNCT) WRITE(6.40) (ITT(1*I)*I=1*5) MINIMUM TRANSMISSIJN TIME (48) ** STATISTICS: *//	#2	FIND FILLED LINE LENGTH						
FIND MINIMUM LINE LENGTH ITT(IVI)=MINO(LENVITT(IVI)) FIND MAXIMUM LINE LENGTH ITT(2, I)= MAXO(LEN.ITT(2,I)) FIND SUM OF LENGTHS SUM=SUMFLDAT(LEN) SUMSQ=SUMSJ+(FLDAT(LEN))**2 100 CONTINUE FIND SAMPLE MEAN AND STANCARD DEVIATION STI(1, I)= SUM/FLDAT(INLNCT) STI(2, I)= S)RT((SUMSQ-(SUM**2)/FLOAT(INLNCT))/FLOAT(INLNCT) WRITE(6,400)(ITT(1,I),I=1,5) 400 FDRMAT(*'O MINIMUM TRANSMISSION TIME (48) *' LENGTH OMS 5 MS 10 MS 20 MS *' STATISTICS:*// WRITE(6,410)(ITT(2,1),I=1,5) 410 FORMAT(*' MAXIMUM*,IOX,5(19)//) WRITE(6,420)(STT(1,I),I=1,5) 420 FDRMAT(*' MAXIMUM*,IOX,5(18)//) WRITE(6,420)(STT(1,I),I=1,5) A20 FDRMAT(*' SAMPLE MEAN',9X,5(F8,2)//)	000 000 000 000 000 000 000 000	LEN=MAXO(LENGTH(J),MTT(I)) IF(DIAG) WRITE(8.50) LEN)					
FIND MAXIMUM LINE LENGTH ITT(2,1) = MAXO(LEN, ITT(2,1)) FIND SUM OF LENGTHS SUMSQ=SUMSJ+(FLDAT(LEN)) **2 100 CONTINUE FIND SAMPLE MEAN AND STANDARD DEVIATION STT(1,1) = SUM/FLDAT(INLNCT) STT(2,1) = SIRT((SUMSQ-(SUM**2)/FLDAT(INLNCT))/FLDAT(INLNCT) WRITE(6,400)(ITT(1,1),1=1,5) 400 FDRMAT(*'O	000 000 000 000 000 000 000 000 000 00	50 FORMAT (18)						0 00
FIND MAXIMUM LINE LENGTH ITT(2, 1) = MAXO(LEN, ITT(2,1)) FIND SUM OF LENGTHS SUMSQ=SUMSJ+(FLOAT(LEN)) **2 100 CONTINUE FIND SAMPLE MEAN AND STANDARD DEVIATION STT(1, 1) = SUM/FLOAT(INLNCT) STT(2, 1) = SIRT((SUMSQ-(SUM**2)/FLOAT(INLNCT))/FLUAT(INLNCT)) WRITE(6,4 0.0) (ITT(1,1),1=1,5) 400 FDEMART(*' CODED LINE*/ *' LENGTH OMS 5 MS 10 MS 20 MS *' STATISTICS:*// *' STATISTICS:*// WRITE(6,4 10) (ITT(2,1),1=1,5) 410 FORMAT(*' MAXIMUM*,10X,5(19)//) WRITE(6,420) (STT(1,1),1=1,5) 420 FDEMAT(*' SAMPLE MEAN*,9X,5(F8,2)//)	100	FIND MINIMUM LINE LENGTH						000
ITT(2.1) = MAXO(LEN.ITT(2.1)) FIND SUM OF LENGTHS SUM=SUMOFLIAT(LEN) SUMSQ= SUM SD+(FLDAT(LEN)) ** 2 100 CONTINUE FIND SAMPLE MEAN AND STANDARD DEVIATION STT(1.1) = SUM/FLOAT(INLNCT) STT(2.1) = S)RT((SUMSQ-(SUM**2)/FLOAT(INLNCT))/FLOAT(INLNCT) WRITE(6.4 0.0) (ITT(1.1).1=1.5) 400 FDRMAT(*'O	000 000 000 000 000 000 000 000 000 00		} }					
ITT(2, 1)= MAXO(LEN, ITT(2,1)) FIND SUM OF LENGTHS SUM=SUMOFL)AT(LEN) SUMSO= SUM SD+(FLDAT(LEN))+2 100 CONTINUE FIND SAMPLE MEAN AND STANDARD DEVIATION STT(1, 1)= SUM/FLDAT(INLNCT) STT(2, 1) = S)RT((SUMSQ-(SUM+2)/FLDAT(INLNCT))/FLDAT(INLNCT) WRITE(6,400)(ITT(1,1),1=1,5) 400 FDRMAT(4'0 MINIMUM TRANSMISSION TIME (48) 4' CODED LINE'/ O MS 5 MS 10 MS 20 MS 4' STATISTICS:'// WRITE(6,410)(ITT(2,1),1=1,5) 410 FORMAT(4' MAXIMUM',10X,5(18)//) WRITE(6,420)(STT(1,1),1=1,5) 420 FDRMAT(4' MAXIMUM',10X,5(18)//) WRITE(6,420)(STT(1,1),1=1,5) 420 FDRMAT(4' MAXIMUM',10X,5(18)//) WRITE(6,420)(STT(1,1),1=1,5) 420 FDRMAT(4' SAMPLE MEAN',9X,5(F8,2)//)	000 000 000 000 000 000 000 000 000 00	FIND MAXIMUM LINE LENGTH						
FIND SUM OF LENGTHS SUM-SUM-SUM SJ+(FLDAT(LEN)) **2 100 CONTINUE FIND SAMPLE MEAN AND STANDARD DEVIATION STT(1, I)= SUM/FLOAT(INLNCT) STT(2, I)= SJRT((SUMS Q-(SUM**2)/FLOAT(INLNCT))/FLUAT(INLNCT) WRITE(6,400)(ITT(1,1),I=1,5) 400 FDRMAT(*'0	000 000 000 000 000 000 000 000 000 00))					
### SUMS SUMS SUMS SUMS SUMS SUMS SUMS S	#2 000 RD DEVIATION 000 **2)/FLOAT(INLNCT))/FLUAT(INLNCT-1)) 000 **2)/FLOAT(INLNCT))/FLUAT(INLNCT-1)) 000 MINIMUM TRANSMISSION TIME (4800 BPS)*//000 000 000 000 000 000 000 000 000							000
SUMSO= SUM SO+(FLDAT(LEN)) **2 100 CONTINUE FIND SAMPLE MEAN AND STANCARD DEVIATION STT(1, I)= SUM/FLOAT(INLNCT) STT(2, I)= S)RT((SUMSQ-(SUM**2)/FLOAT(INLNCT))/FLUAT(INLNCT) WRITE(6,400)(ITT(1,1),I=1,5) ### ### ### ########################	#2 000 000 000 000 000 000 000 000							000
FIND SAMPLE MEAN AND STANCARD DEVIATION STT(1, 1)= SUM/FLOAT(INLNCT) STT(2, 1) = S)RT((SUMS Q-(SUM**2)/FLOAT(INLNCT))/FLUAT(INLNC 300 CONTINUE WRITE(6,400)(ITT(1,1),1=1,5) #00 FDRMAT(#10 MINIMUM TRANSMISSION TIME (46 #1 CODED LINE*/ O MS 5 MS 10 MS 20 MS #1 STATISTICS:*// WRITE(6,410)(ITT(2,1),1=1,5) #10 FDRMAT(#1 MAXIMUM',[0X,5(18)//) WRITE(6,420)(STT(1,1),1=1,5) #20 FDRMAT(#1 SAMPLE MEAN',9X,5(F8,2)//)	##2)/FLOAT(INLNCT))/FLUAT(INLNCT-1)) 000 ##2)/FLOAT(INLNCT))/FLUAT(INLNCT-1)) 000 000 000 MINIMUM TRANSMISSION TIME (4800 BPS)*//000 000 000 000 000 000 000 000 000 0	SUMSQ= SUN SJ+(FLDAT(LEN))+4	* 2					000
STT(1, I)= SUM/FLOAT(INLNCT) STT(2, I)= S)RT((SUMSQ-(SUM**2)/FLOAT(INLNCT))/FLGAT(INLNC 300 CONTINUE WRITE(6.400)(ITT(1,1), I=1.5) 400 FDRMAT(*'0	000 000 000 000 000 000 000 000							
STT(2.1) = S)RT((SUMSQ-(SUM+*2)/FLOAT(INLNCT))/FLOAT(INLNC 300 CONTINUE WRITE(6.4 0.0) (ITT(1.1).I=1.5) 400 FDRMAT(*'O	000 **2)/FLOAT(INLNCT))/FLUAT(INLNCT-1)) 000 000 000 000 MINIMUM TRANSMISSION TIME (4800 BPS)*//000 000 000 000 000 000 000 000 000 0	FIND SAMPLE MEAN AND STANCAR	RD DEVIAT	ION				000
## TE (6.4 0.3) (ITT(1.1).I=1.5) ## TE (6.4 0.3) (ITT(1.1).I=1.5) ## CODED LINE*/ #* LENGTH	000 000 MINIMUM TRANSMISSIDN TIME (4800 BPS) */000 0 MS 5 MS 10 MS 20 MS 40 MS*/ 000 000 000 05) 000 000 000 000 000 000					A T . 1		000
WRITE(6.400)(ITT(1.1).I=1.5) 400 FORMAT(-5) MINIMUM TRANSMISSION TIME (4800 BPS) */000 O MS 5 MS 10 MS 20 MS 40 MS*/ 000 O O O O O O O O O O O O O O O O O	-300 CONTINUE	++2 // FLUA	TINENC	1777760	AILIAEN	C1-1//	
#**O FDEMAT(#*O MINIMUM TRANSMISSIDN TIME (46 #** CCDED LINE*/ #** LENGTH O MS 5 MS 10 MS 20 MS #** STATISTICS:*// #** WRITE(6.410)(ITT(2.1), I=1.5) #** AVAINABLE MEAN*.9X.5(F8.2)//)	MINIMUM TRANSMISSIJN TIME (4800 BPS) "//000 0 MS 5 MS 10 MS 20 MS 40 MS"/ 000 000 000 000 000 000 000 000 000 00		•5)					
#! CCDED LINE!/ #! LENGTH 0 MS 5 MS 10 MS 20 MS #! STATISTICS:!// #RITE(6.410)(ITT(2.1), I=1.5) 410 FORMAT(#! MAXIMUM!.[OX.5(18)//) WRITE(6.420)(STT(1.1), I=1.5) A20 FORMAT(#! SAMPLE MEAN!.9X.5(F8.2)//)	000 0 MS 5 MS 10 MS 20 MS 40 MS' / 000 000 000 000 000 000 000 00	400 FURMATI		TPANSM	ISSIIN	TIME (A	800 BPS1	002
* STATISTICS: *// WRITE (6.410) (ITT (2.1), I=1.5) 410 FORMAT (** MAXIMUM'. [OX.5(18)//) WRITE (6.420) (STT(1.1), I=1.5) A20 FORMAT (** SAMPLE MEAN'.9X.5(F8.2)//)	000 000 000 000 000 000 000 000	** CEDED LINE !/						موم
WRITE(6.410)(ITT(2.1), I=1.5) 410 FORMAT(#* MAXIMUM', [OX.5(18)//) WRITE(6.420)(STT(1.1), I=1.5) A20 FORMAT(** SAMPLE MEAN'.9X.5(F8.2)//)	.5) 000 000 .5) 000 2)//) 000 .5) 003	* STATISTICS: 1//		3 M3	10 M2	20 MS	40 MS	000
** MAXIMUM', [OX, 5(18)//) WEITE(6,420)(STT(1,1),I=1,5) A20 FORMAT(** SAMPLE MEAN',9X,5(F8,2)//)	000 000 000 2)//) 000 (5) 000							
WRITE(6.420)(STT(1.1),I=1.5) A20 FORMAT(* SA MPLE MEAN'.9X.5(F8.2)//)	.5) 000 000 2)//) 000 .5) 003							
* SA PLE MEAN' . 9X . 5 (F8 . 2)//)	2)//) •5) •000	WRITE(6.420)(STT(1.1).I=1	•5)					000
		*1 SAMPLE MEAN . 9X . 5 (F8 . 2	2)//)					000
WRITE(6,430)(STT(2,1),I=1,5) 430 FBFMAT(-430 						
** STAND ARD DEVIATION*.2X.5(F8.2))	(\$ (F8.2)) 000 000		X.5(F8.2))				
RETURN E N D	000	RETURN						000
END OF DEEC UPRINT PROGRAM	COGRAM	END OF DEEC UPRINT PE	ROGRAM		_1¥±5.	PHINIED	=_1615_	
								
								

UNE_ASSIFIED

APPENDIX F

FLOW CHART

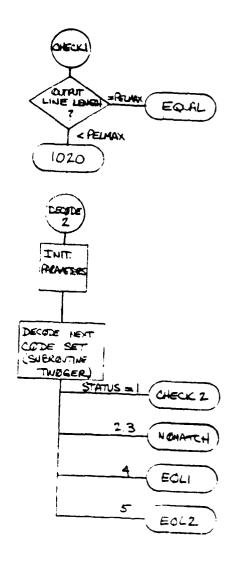
FEDERAL REPUBLIC OF GERMANY

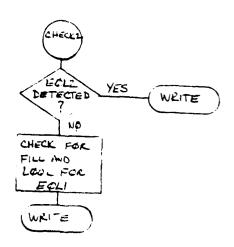


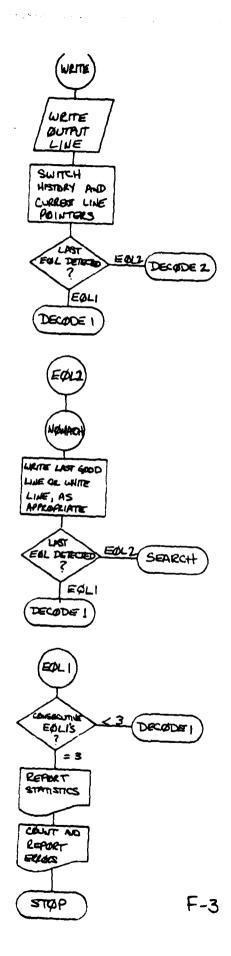
GERMAN

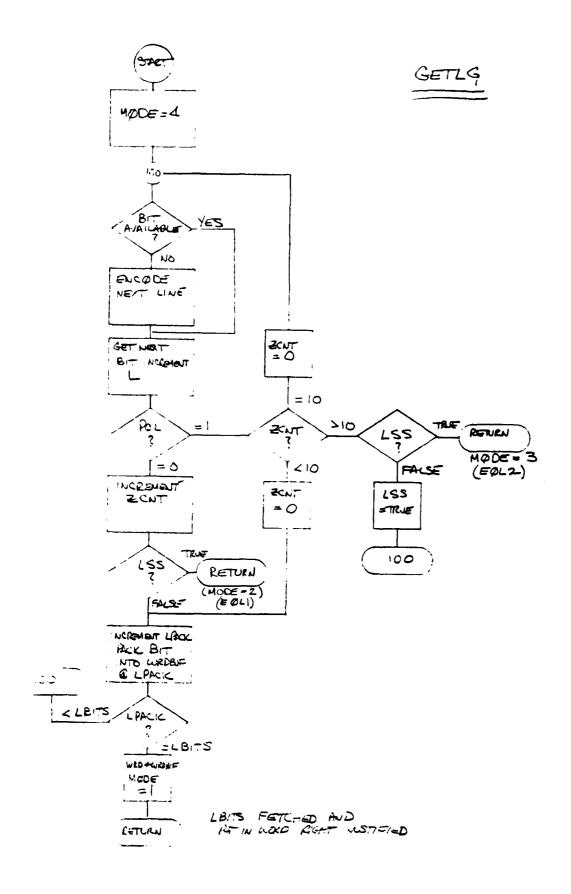
BUTAT LINE & PELMAX DUTPUT LINE TOO LONG OR NO MATCH FOULD IN CODE TRELE

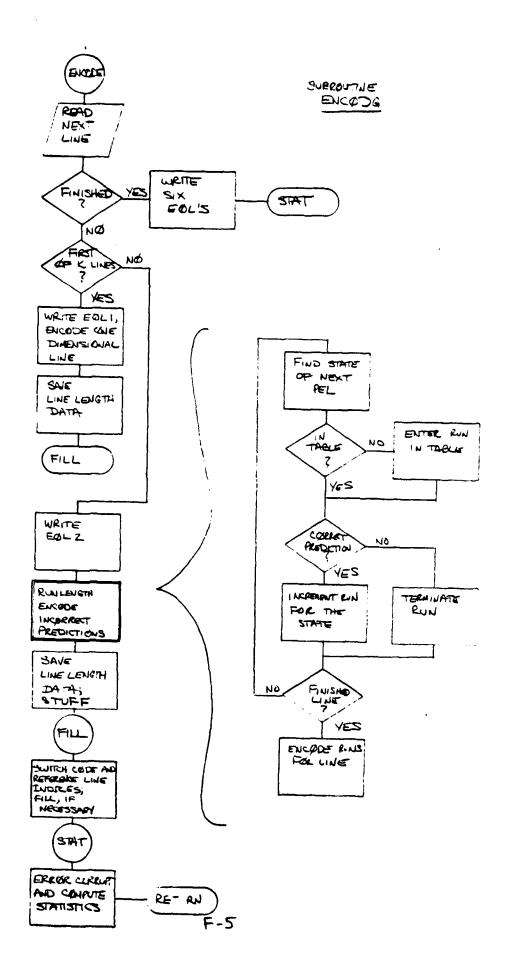
PREHATURE LICE DETECTED

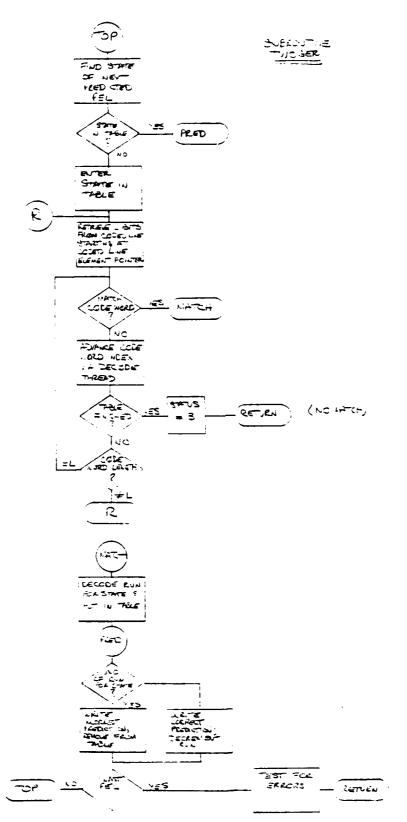












F-6

APPENDIX G

CODE LISTING

FEDERAL REPUBLIC OF GERMANY

JNC_ASSIFIED

INDITIC T. N. T. T. T. S. A. C. C. C. C. T. C. T. S. C. T. C.	PRCGRAM <u>JERMAN</u>	00000
### AL CF3_CF3_ERATE ***********************************		00000
COMMON/JUF PPELBUF (60.2). COMMON JUF PPELBUF (10.2). COMMON JUF PPELBUF (60.2). STERUE (20.0). ERROCTI JUE PPELBUF (60.2). STERUE (20.0). STERUE (20.0). COMMON JUF PREMBUF (20.0). STERUE (20.0).		00000
COMMON / J 32 BIT / MASK (32) , COMASK (132) , COMASK (132) , COMON (15 COMMON / JOHF / PELDUF (60 , 2) , COBUF (240) , COMMON / JOHF / PELDUF (60 , 2) , COBUF (240) , COMMON / JOHF / PELDUF (60 , 2) , COBUF (240) , COMMON / JOHF / PELDUF (60 , 2) , COBUF (240) , COMMON / JOHF / PELDUF (60 , 2) , COBUF (240) , STAT (100) , INPRED (16) , COMMON / JOHF / COS (1) , STAT (16) , STAD (16) , STAD (16) , NON (16) , COMMON / JOHF / COS (1) , STAT (16) , STAD (16) , STAD (16) , NON (16) , COMMON / JOHN / COMMON / COMMON / COMMON / COMMON / COMMON / JOHN	****** LABELED COMMON /G328IT/ ******	00000
INTEGER *ASKs.COMASK,.1817.t.2817 COMMON.PSPEPPELOUF(60.2).COBUF(20.0). GOMMON.PSPEPPELOUF(60.2).STERUE(23.0). GOMMON.PSPEPPELOUF(60.2).STERUE(23.0). **********************************	COMMON 7.328117445K/32). COMASK/32). IRIT (32). 7811 (32)	
COMMON JUFF PELBUF (60.2) COBUF (240) COUNTY (240) COUNTY PELBUF (60.2) COUNTY (240) COUNTY PELBUF (60.2) COUNTY PERBUF (60.2) COUNTY P		00000
COMMON JULY AS JERUE 1201. STRILE 1201. STRILL 10001 COMMON JULY AS JERUE 1201. (STRAT110), STBUF(1728) STRUNC 1228) COMMON JERWY ERROR (1200) COMMON JERWY ERROR (1200) COMMON JERWY ERROR 12000 COMMON JERWY ERROR 12000 COMMON JULY AS JERWAS JERUE 120 COMMON VARIABLES ************************************		00000
COMMON.YSMYCRAGE(19), CODS(3:68:6); PREDOT(16), NPRED(16), 0000 COMMON.YSMYCRAGE(1950) COMMON.YSMYCRAGE(1950) CCOMMON.YSMYCRAGE(1950) INSTANCE (1950)		00000

COMMON_IVERPOOS_CESOOD COMMON	COMMIN MUFF / CUDE(3,92,2), CUDS(3,60,6), PREDCI (10), NPRED(10),	
CCMPCN/FILES/TERY_LPFIC.PELFIL.TEFIL COMMON_IV AR /PELMAX.VRES.EPHASE.CMPMAX.ERRMJO_LINMAX.K COMMON_IV AR /PELMAX.VR.CMPMAX.ERRMJO_LINMAX.K COMMON_IV AR /PELMAX.K COMMON_IV AR /PELMAX.VR.CMPMAX.ERRMJO_LINMAX.K COMMON_IV AR /PELMAX.K COMMON_IV AR /PELMAX.ERRMJO AR /PELMAX.C COMMON_IV		
CCMMON/IV AR/PELMAX.VES.EPHASE.CMPMAX.ERR.DJ.LINMAX.K COMMON/IV AR/PELMAX.VES.EPHASE.CMPMAX.ERR.DJ.LINMAX.K COMMON/IV AR/PELMAX.VES.EPHASE.CMPMAX.ERR.DJ.LINMAX.K COMMON/IV AR/PELMAX.VES.EPHASE.CMPMAX.ERR.DJ.LINMAX.K COMMON/IV AR/PELMAX.VES.EPHASE.CMPMAX.ERR.DJ.LINMAX.K COMMON/IV AR/PELMAX.VES.EPHASE.CMPMAX.ERR.DJ.LINMAX.K COMMON/IC HIM.CT.CONS.ECL.NO.DG.R.C.NT.W.C.P.C.T.VER.D.CT.L.VEN. COMMON/IC HIM.CT.CONS.ECL.NO.DG.R.C.NT.W.C.P.C.T.VEN.DG.R.C.NT.W.C.NT.W.C.NT.W.C.NT.W.C.NT.W.C.NT.W.C.NT.W.C.C.NT.W.C.C.NT.W.C.C.NT.W.C.NT		00000
CCMMCN/ILES/TERS.LEPIL.PETIL.P		00000
COMMON/IV AR/PELMAX.VRES.EPHASE.CNPMAX.ERRMUD.LINMAX.K 0000 COMMON/IV AR/PELMAX.VRES.EPHASE.CNPMAX.ERRMUD.LINMAX.K 0000 COMMON/IV AR/PELMAX.VRES.EPHASE.CNPMAX.ERRMUD.LINMAX.K 0000 COMMON/IV AR/PELMAX.VRES.EPHASE.TNCEP.ERRMUD.LINMAX.K 0000 COMMON/ICHAR/OC.II.MM.TI.NN.YV 0000 CCMMON/ICHAR/OC.II.MM.TI.NN.YV 0000 CCMMON/ICHAR/OC.II.MM.TI.MN.YV 0000 READO(5.11.0) INSW CILINGA-NELII. GD TO 90 0000 COMMON/ICHAR/OC.II.MM.TI.MN.YV 0000 READO(5.11.0) INSW CILINGA-NELII. GD TO 120 0000 COMMON/ICHAR/OC.II.MM.TI.MN.YV 0000 READO(5.11.0) INSW COMMON/ICHAR/OC.II.MM.TI.MN.YV 0000 READO(5.11.0) INSW COMMON/ICHAR/OC.II.MM.TI.MN.YV 0000 READO(5.11.0) INSW COMMON/ICHAR/OC.II.MM.TI.MN.YV 0000 READO(5.11.0) INSW 0000 0000 READO(5.11.0) INSW 0000 0000 0000 0000 00000 000000	CCWMCN/FILES/TERM.LPFIL.PELFIL.CTFIL.EFFIL	00000
COMMON/IV AR /PELMAX.VRES.EPHASE.CMPMAX.ERRMJD.LINMAX.K 0000 CSMMCN/IVAL/TELMOGOTELM		
COMMON/IV AR/PELMAX, VRES, EPHASE, CMPMAX, ERRMUD, LINMAX K COMMON, PV ANT YL NING OF CHANG, OF TE WINDER CELEPY OF LEPYCE 1. * CDELCT: INELCT: CTOATA TO LE RERPINT: ERREFE, ERRLIN, 0000 "COMMON, ICH TO LINKET; COUNSEC, LNNOBE, ZCMT, WR JOUF, PACK, 0000 "COMMON, ICH TO LINKET; CLUCUS CTHEF; TSTFBT 0000 COMMON, ICH TO LARRY COUNTY C		00000
***CSMMCN_PVALY_N_INN_OFFELW_STEED_COLET_S	COMMONZIV ARZPELMAX • VRES•EPHASE•CMPMAX•ERRMUD•LINMAX•K	00000
# ERRCHT, INL.NCT. CONSEC, LNNOBE, ZCYT, #7JBUF, PACK, 0000 CCMMCN/ICHAR/30, II, MM, ITT, NN.YY CCMMCN/ICHAR/30, II, MM, ITT, NN.YY OCCUMEN/ILGIZ/SEABCH, DIAG, SYNC, LSS, WRITE, CHCGL, LINE LOGICAL SEARCH, DIAG, SYNC, LSS, WRITE, CHCGL, LINE OO RITE(6, 100) OO RITE(6, 100) OO READTS, II, ORRESO) INSW IIO POPMAT(18 PARAMETERS: INPUT(=I), CR DEFAULT(=D)?*) OOO READTS, II, ORRESO) INSW III (INSW, EQ. ADD) GO ITO 315 OOO READTS, II, ORRESO) INSW III (INSW, EQ. ADD) GO ITO 315 OOO READTS, II (O) INSW III (INSW, EQ. ANN) GO TO 120 OOO IF (INSW, EQ. ANN) GO TO 120 OOO OF CIT 14 OOO IF (INSW, EQ. ANN) GO TO 120 OOO OF CIT 14 OOO READ WAXIAUM MUMBER OF PELS PER LINE OOO OOO READ WAXIAUM MUMBER OF PELS PER LINE: ') OOO OOO READ WAXIAUM MUMBER OF PELS PER LINE: OOO OOO READTS, II, OOO OOO READTS, II, OO OOO READTS, II, OOO OOO READTS, II, OOO OOO OOO OOO READTS, II, OOO OOO OOO OOO OOO READTS, II, OOO OOO OOO OOO OOO OOO OOO		-00000
TRUCD TREP, CICCUD CITEF, TISTED 0000 CCMMCN/ICHAR/OC, II MN, TITANN, YY 0000 CCMMCN/ICHAR/OC, SYNC, LSS, WRITE, CHCCL, ONE 0000 CCMMCN/ICHAR/OC, SYNC, SYNC, SYNC, LSS, WRITE, CHCCL, ONE 0000 CCMMCN/ICHAR/OC, SYNC, SY	CDELCT: INELCT: TCDATA: TCDEL: ERRPNT: ERRCFF: ERRLIM:	00000
CCMMENN/IGHARY 05.11.NM. TT.NN.YY CCMMENN/IGHARY 05.11.NM. TT.NN.YY CCMMENN/IGHARY 05.11.NM. TT.NN.YY LOGICAL SEARCH. DIAG., SYNC., LSS., WRITE. CHCCL.ONE 0000 FEAD INDET PARAMETERS 0000 FEAD INDET PARAMETERS: (NPUT(=I), CR DEFAULT(=D)?') 0010 FEAT (A) DAMMETERS: (NPUT(=I), CR DEFAULT(=D)?') 0010 100 FORMAT (A) DAMMETERS: (NPUT(=I), CR DEFAULT(=D)?') 110 FORMAT (A) DAMMETERS: (NPUT(=I), CR DEFAULT(=D)?') 0010 FEAD STATE (A) DAMMETERS: (NPUT(=I), CR DEFAULT(=D)?') 1010 FEAD WAXIAUM NUMBER OF PILS PER LINE 0010 1010 1010 FEAD WAXIAUM NUMBER OF PELS PER LINE 0010 1010 1010 FEAD WAXIAUM NUMBER OF PELS PER LINE: ') 0010 1010 FEAD WAXIAUM NUMBER OF PELS PER LINE: ') 0010 1010 FEAD WAXIAUM NUMBER OF PELS PER LINE: ') 0010 READ STATE (A) DAMMETERS: (D) OD		
COMMENTALISTIC SEARCH, DIAG, SYNC, LSS, WRITE, CHCCL, ONE		
LOGICAL SEARCH, DIAG, SYNC, LSS, WRITE, CHCCL, ONE 0000 FEAD IMPLY PARAMETERS 00 WRITE(6,100) 100 FORMAT(12 PARAMETERS: INPUT(=1), CR DEFAULT(=D)?') 0000 110 FORMAT(12) 110 FORMAT(11) 110 FORMAT(11) 111 FORMAT(11) 112 FORMAT(11) 113 FORMAT(15) 115 FORMAT(15) 115 FORMAT(15) 115 FORMAT(15) 115 FORMAT(15) 115 FORMAT(15) 115 FORMAT(15) 116 FORMAT(15) 117 FORMAT(15) 118 FORMAT(15) 119 FORMAT(15) 119 FORMAT(15) 110 INSW 111 FORMAT(15) 111 FORMAT(15) 112 FORMAT(15) 113 FORMAT(15) 114 FORMAT(15) 115 FORMAT(15) 116 CONTINUE 117 FORMAT(15) 117 FORMAT(15) 118 FORMAT(15) 119 FORMAT(15) 119 FORMAT(15) 110 FORMAT(15) 110 FORMAT(15) 110 FORMAT(15) 111 FORMAT(15) 112 FORMAT(15) 113 FORMAT(15) 114 FORMAT(15) 115 FO		
DOOD		00000
### PARAMETERS 0000 100 FORMAT((*) PARAMETERS: INPUT(=I), CR DEFAULT(=D)?') 0000 110 FORMAT(AI) 0000 110 FORMAT(AI) 0000 111 FORMAT(AI) 0000 112 (INSA-PA-DA) GO IN 315 0000 113 FORMAT(AI) 0000 114 WRITE(6:115) 0000 115 FORMAT(*5:IXCAUSTIC PRINTDUT? (Y ER N): ') 0000 116 (INSA-PA-DA) INSW 0000 117 (INSA-PA-DA) GO IN 14 0000 118 FORMAT(*5:IXCAUSTIC PRINTDUT? (Y ER N): ') 0000 119 (INSA-PA-YY) GO IN 14 0000 119 (INSA-PA-YY) GO IN 14 0000 110 GO IC 114 0000 110 GO IC 114 0000 110 FORMAT(*DA) 0000 111 GO IC 114 0000 112 COLDI INUE 0000 113 FORMAT(*DA) 0000 114 FORMAT(*DA) 0000 115 FORMAT(*DA) 0000 116 FORMAT(*DA) 0000 117 FORMAT(*DA) 0000 118 FORMAT(*DA) 0000 119 FORMAT(*DA) 0000 119 FORMAT(*DA) 0000 110 FORMAT(*DA) 0000 111 FORMAT(*DA) 0000 111 FORMAT(*DA) 0000 112 FORMAT(*DA) 0000 113 FORMAT(*DA) 0000 114 FORMAT(*DA) 0000 115 FORMAT(*DA) 0000 0000 116 FORMAT(*DA) 0000 0000 117 FORMAT(*DA) 0000 0000 118 FORMAT(*DA) 0000 0000 119 FORMAT(*DA) 0000 0000 110 FORMAT(*DA) 0000 0000 111 FORMAT(*DA) 0000 0000 112 FORMAT(*DA) 0000 0000 113 FORMAT(*DA) 0000 0000 114 FORMAT(*DA) 0000 0000 115 FORMAT(*DA) 0000 0000 116 FORMAT(*DA) 0000 0000 117 FORMAT(*DA) 0000 0000 118 FORMAT(*DA) 0000 0000 119 FORMAT(*DA) 0000 0000 110 FORMAT(*DA) 0000 0000 110 FORMAT(*DA) 0000 0000 111 FORMAT(*DA) 0000 0000 111 FORMAT(*DA) 0000 0000 111 FORMAT(*DA) 0000 0000 0000 111 FORMAT(*DA) 0000 0000 0000 111 FORMAT(*DA) 0000 0		00000
100 FORWAT(*S PARAMETERS: INPUT(=I), CR DEFAULT(=D)?*) 0000 READ(\$,110, ERRE=90) INSW 1F (INSW.*S_A,DO), GO TO 315 1F (INSW.*S_A,DO), GO TO 316 1F (INSW.*S_A,DO), GO TO 316 1F (INSW.*S_A,DO), GO TO 310 1F (INSW.*S_A,DO), GO TO 320 1F (INSW		-00000
READ(S,110,ERR=90) INSW 0000 110 FORMAT(A1) 0000 15 (INSW-52,DD), GD, TD 315 0000 16 (INSW-52,DD), GD, TD 315 00000 00000 00000 00000 00000 00000 00000 00000 00000		00000
110 FORMATIAL) 11 FORMATIAL) 12 (INSM-P2_DD) GO TO 315 13 (INSM-P2_DD) GO TO 90 11 F (INSM-P2_DD) GO TO 90 12 (INSM-P2_DD) GO TO 90 13 FORMATIFE (5.115) 13 O000 11 F (INSM-C5.115) 13 O000 11 F (INSM-C5.115) 15 FORMATIFE (5.115) 16 CONTINUE 17 CPELMAX.JEJ FARMETER K: ') 18 FORMATIFE (5.115) 19 CONTINUE 10 CONTINUE 10 CONTINUE 11 F (INSM-C5.116) 12 FORMATICAL SAMPLING 13 FORMATICAL SAMPLING 14 F (FELMAX.JEJ) FELMAX.DOOD 15 FORMATICAL SAMPLING 16 CONTINUE 17 FORMATICAL SAMPLING 18 FORMATICAL SAMPLING 19 FORMATICAL SAMPLING 19 FORMATICAL SAMPLING 10 CONTINUE 10 CONTINUE 11 F (FELMAX.JEJ) FELMAX.DOOD 12 FORMATICAL SAMPLING 13 FORMATICAL SAMPLING 14 FORMATICAL SAMPLING 15 FORMATICAL SAMPLING 16 CONTINUE 17 FORMATICAL SAMPLING 18 FORMATICAL SAMPLING 19 CONTINUE 19 CONT		
IF (INS#-PADD) GO TO 315 O0000 READ DIA 903 STIC SWITCH 00000 114 WRITE(6:115) 115 FORMAT ('\$CIACRUSTIC PRINTDUTY (Y ER N): ') READ(5:110) INSW 00000 IF (INS#-EQ.YY) GO TO 116 O0000 GO TO 114 O0000 READ MAXIAUM NUMBER OF PELS PER LINE 00000 120 CONTINUE WRITE(6:130) 130 FORMAT ('\$CIACRUSTIC PRINTDUTY (Y ER N): ') 00000 READ MAXIAUM NUMBER OF PELS PER LINE 00000 120 CONTINUE WRITE(6:130) FORMAT ('\$CIACRUSTIC PRINTDUTY (Y ER N): ') 00000 130 FORMAT ('\$CIACRUSTIC WAXIMUM NUMBER OF PELS PER LINE: ') 00000 WRITE(6:130) FORMAT ('\$CIACRUSTIC WAXIMUM NUMBER OF PELS PER LINE: ') 00000 WRITE(6:150) PELMAX 00000 READ VERTICAL SAMPLING 00000 WRITE(6:170) 100 CONTINUE WRITE(6:170) 100 CONTINUE WRITE(6:191) VRES 100 CONTINUE WRITE(6:191) PEAC (5:180 SRR=160) VRES 00000 PEAC (5:180 SRR=160) VRES 000000 PEA		
IF (INS#.NE.II) GO TO 90 READ SIM GOODT SWITCH 114 WRITE (6.115) 115 FORMAT ('SCIAGNOSTIC PRINTOUTY (Y CR N): ') READ (5.11 0) INSW 10000 IF (INSW.E.G. NN) GO TO 120 GO TO 114 116 CONTINUS DIAG=. TRUE. READ WAXIAUM NUMBER OF PELS PER LINE 20000 120 CONTINUS WRITE (6.130) 130 FORMAT ('SCIAGNOSTIC PRINTOUTY (Y CR N): ') WRITE (6.130) 130 FORMAT ('SCIAGNOSTIC PRINTOUTY (Y CR N): ') WRITE (6.130) 130 FORMAT ('SCIAGNOSTIC PRINTOUTY (Y CR N): ') WRITE (6.130) 130 FORMAT ('SCIAGNOSTIC PRINTOUTY (Y CR N): ') WRITE (6.130) 150 FORMAT ('SCIAGNOSTIC PRINTOUTY (Y CR N): ') WRITE (6.130) 150 FORMAT ('SCIAGNOSTIC PRINTOUTY (Y CR N): ') WRITE (6.130) 150 FORMAT ('SCIAGNOSTIC PRINTOUTY (Y CR N): ') WRITE (6.130) 150 FORMAT ('SCIAGNOSTIC PRINTOUTY (Y CR N): ') WRITE (6.130) READ VERTICAL SAMPLING 150 FORMAT ('SCIAGNOSTIC PRINTOUTY (Y CR N): ') WRITE (6.170) READ VERTICAL SAMPLING 160 CONTINUS WRITE (6.170) WRITE (6.170) WRITE (6.170) WRITE (6.170) IF (VRES-GE-1-AND-VRES-LE-10) GO TO 190 WRITE (6.170) WRITE (6.170) READ PARAMETER K 00000 WRITE (6.192) 102 FORMAT ('SCIAGNOSTIC PRINTOUTY ('SCIAGNOST) COUNTOUTY ('SCIA		
READ DIA 939TIC SWITCH 0000 114 WRITE(6,115) 115 FORMAT (*\$IJAGNUSTIC PRINTDUTY (Y ER N): ') READ(5,110) INSW 0000 IE (INSW,20,YY) GO TO 116 O000 GO TC 114 106 COAT INUE DIAG=, TRUE, 0000 READ MAXIAUM NUMBER OF PELS PER LINE 0000 120 COAT INUE WI ITE(6,130) 130 FORMAT (*\$ZITER PARAMETER K: ') READ VERTICAL SAMPLING 160 COAT INUE 0000 READ VERTICAL SAMPLING 160 COAT INUE 0000 READ VERTICAL SAMPLING 160 COAT INUE 0000 170 FORMAT (*\$ZITER PARAMETER K: ') 0000 READ VERTICAL SAMPLING 180 COAT INUE 0000 READ VERTICAL SAMPLING 180 COAT INUE 0000 180 FORMAT (*\$ZITER PARAMETER K: ') 0000 READ PARAMETER K 0000 READ PARAMETER K 0000 READ PARAMETER K 0000 READ PARAMETER K 0000 READ PARAMETER PARAMETER K: ') READ COAT INUE 0000 READ PARAMETER PARAMETER K: ') 190 COAT INUE 0000 READ PARAMETER PARAMETER K: ') 190 COAT INUE 0000 READ PARAMETER PARAMETER K: ') 190 COAT INUE 0000 READ PARAMETER PARAMETER K: ') 190 COAT INUE 0000 READ PARAMETER PARAMETER K: ') 190 COAT INUE 0000 READ PARAMETER PARAMETER K: ') 190 COAT INUE 0000 READ PARAMETER K 0000 READ PARAMETER K 00000 READ PARAMETER A 00000 READ PARAMETER K 00000 READ PARAMETER A 00000 READ PARAMETER R 00000 R 0000		00000
READ DIA WOJOTIC SWITCH 00000		00000
114 WRITE(6.115) 115 FDRMAT(*2TAGNOSTIC PRINTDUTY (Y CR N): ') READ(\$.110) INSW 10000 IE(INSW.EQ.YY) GO TO 116 OO000 15 (INSW.EQ.NN) GO TO 120 OO000 16 CONTINUE WRITE(6.130) 10 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 10 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 10 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 10 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 10 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 10 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 11 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 12 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 13 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 14 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 15 FORWAT (*0NUMBER OUT OF RANGE (=*.16.*)') 16 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 17 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 18 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 19 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 19 CONTINUE 10 FORWAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: ') 19 CONTINUE 19 CONTINUE 10 CONTINUE 10 CONTINUE 10 CONTINUE 11 FORWAT (*3ENTER PARAMETER K: ') 12 FORMAT (*3ENTER PARAMETER K: ') 13 FORWAT (*3ENTER PARAMETER K: ') 14 FORWAT (*3ENTER PARAMETER K: ') 15 FORWAT (*3ENTER PARAMETER K: ') 16 FORMAT (*3ENTER PARAMETER K: ') 17 FORWAT (*3ENTER PARAMETER K: ') 18 FORWAT (*3ENTER PARAMETER K: ') 19 CONTINUE 10 CONTINUE 11 FORWAT (*3ENTER PARAMETER K: ') 12 FORMAT (*3ENTER PARAMETER K: ') 13 FORWAT (*3ENTER PARAMETER K: ') 14 FORWAT (*3ENTER PARAMETER K: ') 15 FORWAT (*3ENTER PARAMETER K: ') 16 FORWAT (*3ENTER PARAMETER K: ') 17 FORWAT (*3ENTER PARAMETER K: ') 18 FORWAT (*3ENTER PARAMETER K: ') 19 CONTINUE WRITE(6.150) K OO000 GO TO 190 OO000		-00000
To promate (1.9) To p		
READ(5:110) INSW 1F(INSW.EQ.YY) GD TO 116 O0000 IF(INSW.EQ.YY) GD TO 116 O0000 OT C 114 O0000 DI AG TC 114 O0000 READ MAXIMUM NUMBER OF PELS PER LINE O0000 WRITE(6:130) 130 FORMAT('\$ \$6\TER MAXIMUM NUMBER OF PELS PER LINE: ') O0000 140 FORMAT('\$ \$6\TER MAXIMUM NUMBER OF PELS PER LINE: ') O0000 IF(FEL MAX.\$ \$6\LAND.\$ PELMAX O0000 IF(FEL MAX.\$ \$6\LAND.\$ PELMAX.\$ \$1.728) GO TO 100 O0000 OF FORMAT('ON) MER OUT OF RANGE (='.16.')') O0000 READ VERTICAL SAMPLING O0000 READ VERTICAL SAMPLING O0000 READ VERTICAL SAMPLING O0000 OF FEMAT('SENTER VERTICAL SAMPLING: ') O0000 O0000 OF FEMAT('SENTER VERTICAL SAMPLING: ') O0000 ON0000 ON00000000000000000000		
TE(INSW.EQ.YY) GT TO 116		
IF (INSweed.NN) GD TD 120		00000
GO TC 114 116 CENT INUE DIAG=: TRUE: READ MAXIAUM NUMBER OF PELS PER LINE 120 CONTINUE WRITE(6,130) 130 FORMAT(14) SAITER MAXIMUM NUMBER OF PELS PER LINE: ') 140 FORMAT(14) O000 140 FORMAT(14) PELMAX 150 FORMAT(14) PELMAX 150 FORMAT(14) PELMAX 150 FORMAT(14) PELMAX 150 FORMAT(14) O000 READ VERTICAL SAMPLING 160 CONTINUE WRITE(6,15) PELMAX 170 FORMAT(15-INDER VERTICAL SAMPLING: ') READ VERTICAL SAMPLING 160 CONTINUE WRITE(6,150) VRES 170 FORMAT(15-INDER VERTICAL SAMPLING: ') REAC(5,180,5RR=160) VRES 180 FORMAT(12) 1F(VRES-6E-1-ANDAVRES-LE-10) GO TC 190 WRITE(6,150) VRES 190 CONTINUE WRITE(6,150) K 190 CONTINUE WR		00000
DIAS== TRUE	GD TC 114	00000
READ WAXIAUM NUMBER OF PELS PER LINE 120 COATINUE WRITE(6.130) 130 FORMAT('SATTER MAXIMUM NUMBER OF PELS PER LINE:') 0000 140 FORMAT(14) 1F(PELMAX.SE.120) PELMAX 0000 WRITE(6.153) PELMAX 150 FORMAT('ONUMBER DUT OF RANGE (='.16.')') 0000 READ VERTICAL SAMPLING 0000 READ VERTICAL SAMPLING 160 CONTINUE WRITE(6.170) 170 FORMAT('SANTER VERTICAL SAMPLING:') 180 FORMAT(I2) 1F(VRES.GE.1.AND.VRES.LE.10) GD TC 190 WRITE(6.150) LYRES 0000 READ PARAMETER K 0000 READ PARAMETER K: ') READ PARAMETER K: ') READ PARAMETER RESENCE SOLUTION RESENCE SOLU		
READ MAXIAUM NUMBER OF PELS PER LINE 120 CONTINUE WRITE(6,130) 130 FORMAT(1'S-ATTER MAXIMUM NUMBER OF PELS PER LINE:') 140 FORMAT(1'S-ATTER MAXIMUM NUMBER OF PELS PER LINE:') 140 FORMAT(14) 150 FORMAT(14) 150 FORMAT(10 NUMBER OUT OF RANGE (='.16.')') 150 FORMAT(10 NUMBER OUT OF RANGE (='.16.')') 150 FORMAT(10 NUMBER OUT OF RANGE (='.16.')') 160 CONTINUE WRITE(6,170) 170 FORMAT(12) 180 FORMAT(13) 180 FORM		
120 CONTINUE	READ WAXIAUM NUMBER OF PELS PER LINE	
#RITE(6.130) 130 FORMAT('\$2NTER MAXIMUM NUMBER OF PELS PER LIN::') 0000 READ(5.140.FRR=120) PELMAX 140 FORMAT(14) 0000 WRITE(6.15J) PELMAX 150 FORMAT('0NUMBER OUT OF RANGE (=*.16.*)') 0000 READ VERTICAL SAMPLING 0000 READ VERTICAL SAMPLING 160 CONTINUE WRITE(6.170) 170 FURMAT('3ENTER VERTICAL SAMPLING:') REAC(5.180.SRR=160) VRES 180 FORMAT(12) 0000 WRITE(6.150) VRES 0000 WRITE(6.150) VRES 0000 READ PARAMETER K 00000 READ PARAMETER K 00000 PEMAT ('\$ENTER PARAMETER K: ') REAC(5.140.SRR=190) K 00000 PEMAT ('\$ENTER PARAMETER K: ') REAC(5.140.SRR=190) K 00000 PEMAT ('\$ENTER PARAMETER K: ') REAC(5.150.SRR=190) K 0000000 PEMAT ('\$ENTER PARAMETER K: ') REAC(5.150.SRR=190) K 00000 PEMAT ('\$ENTER PARAMETER K: ') REAC(5.150.SRR=190) K 000000 PEMAT ('\$ENTER PARAMETER K: ') REAC(5.150.SRR=190) K 00000 PEMAT ('\$ENTER PARAMETER K: ')	READ WAXIOUS NOWDER OF FEED FER LINE	00000
130 FORMAT (*3ENTER WAXIMUM NUMBER OF PELS PER LINE: *) 140 FORMAT (14) 150 FORMAT (14) 150 FORMAT (15) PELMAX 150 FORMAT (*ONJMBER DUT OF RANGE (=*.16.*)*) 160 CONT INUE WRITE (6.1 70) 170 FORMAT (*SENTER VERTICAL SAMPLING: *) 180 FORMAT (*SENTER VERTICAL SAMPLING: *) 191 FORMAT (*SENTER VERTICAL SAMPLING: *) 192 FORMAT (*SENTER VERTICAL SAMPLING: *) 194 CONTINUE 195 CONTINUE 196 CONTINUE 197 CONTINUE 198 CONTINUE 199 CONTINUE 190 CONTINUE 190 CONTINUE 191 CONTINUE 192 FORMAT (*SENTER PARAMETER K: *) 193 CONTINUE 194 FORMAT (*SENTER PARAMETER K: *) 195 CONTINUE 197 FORMAT (*SENTER PARAMETER K: *) 198 CONTINUE 199 CONTINUE 190 CONTINUE 190 CONTINUE 191 FORMAT (*SENTER PARAMETER K: *) 190 CONTINUE 190	120 CONT INUE	_00000
READ(5*1+0*CRR=120) PELMAX		00000
140 FDRMAT(14) IF (PELMAX.jE.1.AND.PELMAX.LE.1728) GD TD 160 O000 WRITE(6.153) PELMAX 150 FDRMAT('ONUMBER DUT OF RANGE (='.16.')') GD TC 120 READ VERTICAL SAMPLING 160 CDNTINUE WRITE(6.170) 170 FDRMAT('SENTER VERTICAL SAMPLING:') REAC(5.180.SER=160) VRES 180 FDRMAI(12). IF (VRES-GE-1.AND.VRES-LE.10) GD TC 190 WRITE(6.150) (VRES 0000 READ PARAWETER K 0000 READ PARAWETER K 0000 READ PARAWETER K 0000 190 CONTINUE WRITE(6.192) 192 FDRMAT('SENTER PARAMETER K:') READ(5.140.ERR=190) K 0000 WRITE(6.150) K 0000 WRITE(6.150) K 0000 O000 O000 O000 O0000 O00000 O0000 O0000 O0000 O0000 O0000 O0000 O0000 O0000 O00000 O0000 O00000 O0000 O0000 O0000 O0000 O0000 O0000 O0000 O0000 O00000 O0000 O0000 O0000 O0000 O0000 O0000 O0000 O0000 O0000		
IF (PELMAX . JE.1 . AND . PELMAX . LE.1728) GD TD 160		
##ITE(6,153) PELMAX 150 FCRMAT ('0NJMBER DUT OF RANGE (='.16.')') GO IC 120 READ VERTICAL SAMPLING 160 CDNT INUE ##ITE(6,170) 170 FORMAT ('\$ENTER VERTICAL SAMPLING: ') FE AC (5,180,5RR=160) VRES 180 FORMAT (12) IF (VRES-GE-1-AND-VRES-LE-10) GD TC 190 ##ITE(6,150) LVRES 0000 READ PARAMETER K 0000 READ PARAMETER K 0000 190 CONTINUE ##ITE(6,192) 192 FORMAT ('\$ENTER PARAMETER K: ') READ (5,140,5RR=190) K 0000 ##ITE(6,150) K 0000 0000 ##ITE(6,150) K 0000 0000 ##ITE(6,150) K 00000 0000 ##ITE(6,150) K 00000 00000 00000 00000 00000 0000		
150 FCFMAT ('ONUMBER DUT OF RANGE (=*,16,*)*) GO TC 120 READ VERTICAL SAMPLING 160 CONTINUE WRITE(6,170) 170 FURMAT ('\$ ENTER VERTICAL SAMPLING: ') READ SERRE 160) VRES 180 FORMAT (12) IF (VRES-GE-1-AND-VRES-LE-10) GO TC 190 WRITE(6,150) LVRES CO TO 160 READ PARAMETER K 190 CONTINUE WRITE(6,192) 192 FOFMAT ('\$ ENTER PARAMETER K: ') READ (5,140,ERR=190) K 1F (Wassel-1-AND-VRES-LE-10) GO TO 200 WRITE(6,192) 192 FOFMAT ('\$ ENTER PARAMETER K: ') READ (5,140,ERR=190) K 1F (Wassel-1-AND-VR-VET-3000) GO TO 200 WRITE(6,150) K 00000 GO TO 190 00000		00000
READ VERTICAL SAMPLING 160 CONTINUE WRITE(6,170) 170 FURMAT('VENTER VERTICAL SAMPLING: ') REAC(5,180,5RR=160) VRES 180 FORMAI(12) IF (VRES-GE-1-AND-VRES-LE-10) GD TC 190 WRITE(6,150) VRES CO TO 160 READ PARAMETER K 190 CONTINUE WRITE(6,192) 192 FORMAT('\$=:TER PARAMETER K: ') READ(5,140,ERR=190) K READ(5,140,ERR=190) K OOOO WRITE(6,150) K OOOO GO TO 190 OOOO GO TO 190 OOOOO GO TO 190		00000
160 CDNT INUE WRITE(6.170) 170 FURMAT('\$ENTER VERTICAL SAMPLING:') REAC(5.180.5RR=160) VRES 180 FURMAT(12) IF (VRES.6E.1.AND.VRES.LE.10) GD TC 190 WRITE(6.150) LVRES 0000 READ PARAMETER K 00000 READ PARAMETER K 00000 URITE(6.192) 190 CONTINUE WRITE(6.192) 192 FURMAT('\$ENTER PARAMETER K: ') READ(5.140.6RR=190) K 0000 READ(5.140.6RR=190) K 0000 WRITE(6.150) K 0000 GD TO 190 0000		00000
160 CDNT INUE WRITE(6.170) 170 FURMAT('\$ENTER VERTICAL SAMPLING:') REAC(5.180.5RR=160) VRES 180 FURMAT(12) IF (VRES.6E.1.AND.VRES.LE.10) GD TC 190 WRITE(6.150) LVRES 0000 READ PARAMETER K 00000 READ PARAMETER K 00000 URITE(6.192) 190 CONTINUE WRITE(6.192) 192 FURMAT('\$ENTER PARAMETER K: ') READ(5.140.6RR=190) K 0000 READ(5.140.6RR=190) K 0000 WRITE(6.150) K 0000 GD TO 190 0000	READ VERTICAL SAMPLING	
160 CDNT INUE WRITE(6,170) 170 FURMAY(**SENTER VERTICAL SAMPLING: *) FEAC(5,180.SER=160) VRES 180 FURMAT(12). IF (VRES.GE.1.AND.VRES.LE.10) GD TC 190 WRITE(6,150) LVRES 0000 READ PARAMETER K 0000 READ PARAMETER K 0000 WRITE(6,192) 190 CONTINUE WRITE(6,192) 192 FURMAT(**SENTER PARAMETER K: *) READ(5,140.ERR=190) K 0000 WRITE(6,150) K 0000 GD TO 190 0000	READ VERTICAL SAMPLING	
WRITE(6.170) 170 FURMAT(*\$ENTER VERTICAL SAMPLING: *) FEAC(5.180.\$ER=160) VRES 180 FURMAT(12). 1F(VRES.GE.1.AND.VRES.LE.10) GD TC 190 WRITE(6.150) VRES 0000 READ PARAMETER K 0000 READ PARAMETER K 0000 WRITE(6.192) 190 CONTINUE WRITE(6.192) 192 FURMAT(*\$ENTER PARAMETER K: *) READ(5.140.\$ER=190) K 0000 WRITE(6.150) K 0000 GD TO 190 0000 0000	160 CONTINUE	
170 FURMAT("SENTER VERTICAL SAMPLING: ")		
REAC (5 , 180, ERR=160) VRES 180 FORMAT (12). IF (VRES , GE , 1 - AND , VRES , LE , 10) GD TC 190 WRITE (6, 150) , VRES 0000 READ PARAMETER K 0000 190 CONTINUE WRITE (6, 192) 192 FORMAT (* \$ \in \text{TER PARAMETER K: '}) READ (5, 140, \in \text{ERR} = 190) K WRITE (6, 150) K 0000 GD TO 190 0000 0000		00000
IF (VRES.GE.1.AND.VRES.LE.10) GD TC 190 WRITE(6.150) VRES O000 READ PARAMETER K O000 190 CONTINUE WRITE(6.192) 192 FORMAT (*5:ITER PARAMETER K: *) READ(5.140.ERR=190) K O000 WRITE(6.150) K O000 GD TO 190 O000 O000		00000
#RITE(6:150); VRES 0000 #READ PARAMETER K 0000 190 CONTINUE 0000 #RITE(6:192) 0000 192 FORMAT (****E-TER PARAMETER K: *) 0000 READ(5:140:ERR=190) K 0000 #RITE(6:150) K 0000 GD TO 190 0000	180 FORMAT(12).	_0000
### ### ##############################	IF (VRES-GE-1-AND-VRES-LE-10) GD TC 190	
READ PARAMETER K 0000 190 CONTINUE WRITE(6.192) 192 FORMAT ('\$ENTER PARAMETER K: ') READ(5.140.ERR=190) K 0000 READ(5.140.ERR=190) K 0000 WRITE(6.150) K 0000 GD TO 190 0000	WRITE(6,150) 1 VRES	
190 CONTINUE 0000 WRITE(6:192) 0000 192 FORMAT ('\$EITER PARAMETER K: ') 0000 READ(5:140:ERR=190) K 0000 IF (K:GE:1*AND*K*LE*3000) GD TO 200 0000 WRITE(6:150) K 0000 GD TO 190 0000		0000
190 CONTINUE 0000 WRITE(6:192) 0000 192 FORMAT ('\$EITER PARAMETER K: ') 0000 READ(5:140:ERR=190) K 0000 IF (K:GE:1*AND*K*LE*3000) GD TO 200 0000 WRITE(6:150) K 0000 GD TO 190 0000	READ PARAVETER K	00000
#RITE(6.192) 192 FORMAT (*\$ENTER PARAMETER K: *) READ(5.140.ERR=190) K O000 IF (K.GE.1*AND****=13000) GO TO 200 WRITE(6.150) K O000 GD TO 190 O000		0000
192 FORMAT ('\$E:ITER PARAMETER K: ') READ(5,140,ERR=190) K 17 (K:GE:1:AND:K:E:3000) GO TO 200 WRITE(6,150) K 0000 GD TO 190 0000		00000
READ(5.140.ERR=190) K 0000 IF (K.4G5.1.4AND.K.4E5.3000) GO TO 200 0000 WRITE(6.150) K 0000 GD TO 190 0000		00000
#RITE(6,150) K 0000 0000 0000 0000 0000 0000 0000	WR ITE(6.192)	
WRITE(6.150) K 0000 GD TO 190 0000	WRITE(6.192) 192 FORMAT('\$ENTER PARAMETER K: ')	00000
GD 10 190 0000	WRITE(6,192) 192 FORMAT('\$ENTER PARAMETER K: ') READ(5,140,ERR=190) K	
0000	WRITE(6,192) 192 FOFMAT ('\$ = ITER PARAMETER K: ') READ(5,140,ERR=190) K 1F (K\$65;1 *AND*K *L5*3000) GO TO 200	-000 (x
READ ERROR PAITERN PHASE 0000	WRITE(6.192) 192 FORMAT('\$ = :ITER PARAMETER K: ') READ(5.140.ER=190) K IF (K*6E*1 *AND*K*LE*3000) GO TO 200 WRITE(6.150) K	0000 0
	WRITE(6.192) 192 FORMAT ('SENTER PARAMETER K: ') READ(5.140.ERR=190) K IF (K:GE:V:ANDVK:LE:3000) GO TO 200 WRITE(6.150) K GD TO 190	00000 00000 00000

200 CCNTINUE	30000
WRITE(6,210)	000005
210 FORMAT (SENTER ERROR PATTERN PHASE: 1)	300008
READ(5.22 0.ERR=200) EPHASE	000008
IF(EPHASE .GE.O .AND.EPHASE.LE.3) GO TO 240	
WRITE(6.150) EPHASE	
GD TO 200	00000\$
C READ MINIMUM COMPRESSED LINE LENGTH	000005
C READ MINIMUM COMPRESSED LINE LENGTH	000005
240 CONTINUE	000005
250 FORMAT ('SENTER MINIMUM COMPRESSED LINE LENGTH: 1)	000005
READ(5.140.ERR=240) CYPMAX IF (CMPMAX.GE.O.AND.CMPMAX.LE.1728) GO TO 320	200000
WRITE(6.150) CMPMAX	000005
GO TO 240	000010
<u>C</u>	000010
C READ NUMBER OF SCAN LINES TO BE PROCESSED	000010
WOITE(A. 330)	000010
330 FORMAT (SNILVARED OF SCAN I INFS TO BE DECCESSED = 2 4)	000016
MENDIALIZATION CINNAX	000010
IF(LINMAX.GE.1.AND.LINMAX.LE.300C) GD TO 280 WRITE(6.150) LINMAX	000010
GO TO 320	000010
c	000011
C READ EFRER 40 DE	
200 (54 7 1)	000011
280 CCNTINUE WRITE(6,290)	000011
290 FORMAT ("\$ ERROR MODE= ? (M=MANUAL , T= TAPE , N=NO _ nkors) ")	000011
READ(5.11 0.ERR=290) EREMOD	00001
IF(ERRMOD =EQ = MM) GO TO 300	000011
IF (ERRMO) .EQ.TT) GO TO 315	00001:
GD TO 350	00001
C READ ERROR LICATIONS	00001.
	000012
C 300 CONTINUE	00001
ERRL IM=1	00001
305 READ(5,140) ERRORS(ERRLIM)	00001.
1F(ERRORS(ERRL IM)*EQ*9999) 60 TO 310	
ERRLIM#SRRLIM+1 GD TO 305	00001.
310 รับหา้าหนัร	00001.
ERRLIM=ERRLIM-1	00001.
GO_TO_350	00001.
C READ ERROR TAPE FILE AND OPEN	00001.
THE PILE AND UPEN	00001.
315 CONTINUE	00001
C	00001.
ERAL IMEL PEAD(3.11 8.5N)-117) EPPCS(COULTY)	00001.
READ(3.318.END=317) ERRORS(ERRLIM) ERRI IM=FRRLIM+1	00001
316 READ (3.319.END=317) ERRORS (ERRLIM)	00001
318 FCRMAT(I16)	000014
ERRORS (ENRLIM) = ERRORS (ERRLIM) + ERRORS (ERRLIM-1)	
ERRLIM=ERRLIM+1 GO TO 316	00001- 00001-
317 ERRETMEERRETMET	00001
C	00001
_ 350 CONTINUE	00001
C 360 CENTINUE	00001- 00001:
SOVERTINOS PARAMETERS	
C	00001:
WRITE(6+400) PELMAX VRES & CHPHAS E CMPMAX LINNAX	00001
# "JAKANAT("11 NPUT PARAMETERS!"/ * "JAKANAM NUMBER OF PELS POR LINE=".16/	00001
* OVERTICAL SAMPLING: NET - 14/	00001
OPARAMETER K = ', 14/	00001
# * *JERROR PATTERN PHASE = *. 14/	00001
TOWN NIME COMPRESSED LINE LENGTH TIET DITS!	
* 'ONUMBER OF SCAN LINES TO BE PROCESSED = '.16) IE(EREMOD.EQ.NN) WRITE(6.410)	00001
410 FORMAT ('ONO ERRORS INSERTED')	00001
IF (ERRWOD .= G.MM) WRITE (6,140) (ERRORS (1), I=1, ERRLIM)	00001

بغفيف	IFTERNINGS: UG TT) WRITE(6:420) ERHLIM D FORMAT(I12: " ERRORS OBTAINED FROM ERROR TAPE") ARREST ARREST BEGIN PROGRAM ************************************	
C C	init iali ze	0000
	TCCEL=0	00001
	TCDATA=0 ERRPNT=1	0000
	ERRCNT=0	0000
	INLNCTED	0000
	ERROFF=E> HA SE + 1024 CDELCT = 32	0000
	OTEL Pa 1	- 0000
	CDELF= 32+1 CDNSEC =1	0000 0000
	INREF= I	0000
	INCCD=2	0000
	CT COC= 2	0000
C		0000
	CO 000 I= 1+2+0 STFBUF(I) =0	9999
	CDBUF(I)=0	0000
80	0 CONTINUE DD 850 I=1.60	9000
	OT BUF(1.) TR SF)=0.	0000
	CTBUF(I,JTCCD)=0	0000
	PELBUF(I.INREF)=0 PELBUF(I.INCO)=0	
85	O CONTINUE	0000
	SEARCH= •TRUE • SYNC=•FALSE •	0000
	WR I TE= oF4 LSE o	0000
<u></u>		
	SEARCH MODE: LOCK FOR EOL1 BIT-BY-BIT	0000
. 900	- CENT INUE	
	L=C	0000
	LSS= •FALSE• ZCNY=0	0000
	WROB UF =0	0000
	LPACK=0 CALL GETLG(13,MODE,LBITS,L)	0000
	GC TO (910.930,930,920),MODE	0000
910	CONTINUE	0000
	CONTING	0000
<u>c</u> C	ECL NCT FOUND; ADVANCE PCINTER AND TRY AGAIN	0000
C	.CDFLF=CDFL9+1	2000
	GO TO 900	0000
9 20	CONT INUE	0000
930		0000
<u>c </u>		0000
č	EOLI FCUND	0000
	SEARCH=AFALSEA	0000
	CDELP=CDELP+L IF(WRITE) GO TO 935	0000
	WRITE-TRUCT	
	GO TO 950	0000
<u>9 35</u>	CONT INUE	0000
č	SET OUTPUT DECODE LINE TO 0 AND WRITE CUT	0000
	00 950 1=1.60 0TBUF(1.3TCOD)=0	0000
9 50	CONT INUE	0000
	WRITE(2) GTLNNOVPELMAX (GTOUF(IVETCED) VI=1V60)	
960	OTLNO=LNDBF CONTINUE	0000
	IF (MODE-2) 955,1000,900	0000
965	STCP 965 O CONTINUE	0000
		0000
<u> </u>	PERFORM ONE-DIMENSIONAL DECODE OF A COMPLETE LINE	0000
e C	FIRST, SET DUTPUT BUFFER TO WHITE (CNLY BLACK RUNS WILL BE INSERTED)	
=		0000
<u> </u>	CO 1010 1=1.60	

TOTO CONTINUE		00002
C INDEX=3		00002
COLOR= 1		00002
CTEL P= 1		00002
LSS= .FALSE.		00002
ZCNT=0		00002 00002
CALL ONEGER	(INDEX. COLUR. STATUS.L)	00002
	-1 070-1 070-1 035-1 040) -STATUS	
STCP 1000	2 3 4 5	00002 00002
C 		
C ALL RUNS ADDE		0000 <i>2</i> 00002
C CONTINUE	· · · · · · · · · · · · · · · · · · ·	00002
CHE - TRUE .	051 1144 1071 1070 1050	00002
1031 CONTINUE	PELMAX1 1031 1032 1050	00002
)LOR=4JD(COLOR+2.2)+1	00002
INDEX-3		90002 00002
GO TO 1020 3000 CONTINUE		00002
	and the second s	00002
C PERFORM TWU-D	DIMENSIONAL DECODE	00002
C		00002
	PUT BUFFER TO WHITE	00002
C C	UNS WILL BE INSERTED)	90002 00002
DO 3010 I = 1		00002
CTEUFITATIO	.'Jō)=0	00002
3010 CONTINUE		00002
CTELF=1		00002
C 		00002
ZCNT=0	,	00002
CALL TWOSER	(INDEX.CCLOR.STATUS.L)	00002
GD TO (30.30 C	1,1070,1070,1035,1040),STAYUS	00002 00002
STOP 3000		00002
C RUN ADDED: LD	NAME OF MENT OFF	00002 00002
RUN ADDED; LU	OOK FOR NEXT RUN	
BUNITADO OEOE		00002
ONE - FALSE.	ا با استنامیا با با در است دارید و از پیشتان می با معتبد با با استان با با استان با با استان با با استان با با	00002
I LINE LENGTHEP	PELMAX: CHECK FOR FILL AND LOOK FOR EUL	.1 00002
		20002
1032 CONTINUE ZERC=-1		00002
LSS- FALSE		00002
ZCNT=0 1033 CONTINUE		£0000 £0000
ZE FC=ZERU+1		20003
WR CRUF=0		00003 00003
L= C	angung pangung ang ang ang ang ang ang ang ang ang a	00003
	(1.MODE.LBITS.L)	00003
60 10 (10 34	1,1060,1060,1050),MODE	
00 10 11034		00003
CHECK FOR FIL		00003
1034 CONTINUE		£0000
c		00003
CDELP= COLLP) fl. }+\$ } -60 -7 0 -1033	00003
IF (ZERO. E.	(0) GO TO 1070	00003
STCP 1 034		00003
C PREMATURE EOL	DETECTED	E0000
C PREMATORE EUL	- DUTGUILD	
C FOLL DETECTED	_	00003
C ECLI DETECTED)	
1035 CONTINUE	.	00003
CDELP=CDeL	<u> </u>	20003 20003
	E-1) CONSEC=CONSEC+1	00003

UNC. ASSIFIED

IF (CCNSEC-271080,1000,2000	00003:
C	00003:
C ECI 2 CETECIAC	00003
1040 CONTINUE	00003.
COELP4CDE LP4L	-00003:
STATUS=5	00003.
C GO YC 1080	00003.
C	00003.
C PREBLENS PROBLES	00003
1050 STOP 1050	00003 00003
	-00003
C LINE LENGTH CORRECT, EDL DETECTED PROPERLY; WRITE OUTPUT LINE	00003
TOSO CONTINCE	00003
COELP=COELP +L	00003
LOG. LEL. (DO) TO ALBERTO). XAM JERONK JULIAN ABOUT OF THE CONTROL	00003
CONSEC=1	00003
IF (ONE) 3 Y YCOUT QUE	-00003
TEMP=0 TREF CT REF=CT COD	00003:
CTCCO=TEAP	00003
IF(MODE,EQ.2) GO TO 1000	00003
GO TO 3030	<u> </u>
C . LINE TOO LONG OR NO MATCH	00003
Control of the contro	00003
1070 CONTINUE WRITE=.FALSE.	00003:
C WASTE-OFACIES	00003
Č LINE SHORT	00003
1 080 CONTINUE	20003
IF(.NOT.SYNC) GD TO 1090	00003
· ************************************	- 00003
C WRITE LAST GOOD LINE	00003
VR (7F12) NYI NNO DEL MAY ACCYPUELLA CYCEEL ACELACIA	00003
SYNC=.FA.SE.	00003
1090 CONT INUE	00003
C C C C C C C C C C C C C C C C C C C	00003 00003
C WRITE A WILTE LINE	00003
C DO 1100 I =1 . 60	00003
1100 OTBUF(1,3 TCO3)=0	<u>00003</u>
WRITE(2) OT LNNO .PEL MAX . (OTBUF(1.OT COE) . I=1.60)	00003
IF(STATUS .EQ .4) GD TO 1000	
SEARCH = TRUE.	00003
90 TO 900	00003
C END CF MESSAGE	00003 00003
	00003
2000 CCNTINUE	00003
2010 FORMAT ("0 END OF MESSAGE DETECTED (".12." EDL''S)")	00003
C	00003
C REPORT COMPRESSION FACTORY ERROR SENSITIVITY FACTOR VOIT ERROR RATE. C	-00003
ERRATE =FLOAT(ERRCNT) /FLCAT(TCDEL)	00003 00003
WRITE(6.2020) TCDEL, TCDATA, TSTERT, INLNCT, ERRATE	00003
2020 FOFMAT ('0 TOTAL NUMBER OF CODED BITS = '.18/	E0000
♦ • • • • • • • • • • • • • • • • • • •	50000
* '0 TO TALL NUMBER OF INPUT LINES PROCESSED = '.I3/	00003
C ************************************	- 00003 00003
CALL STATS(STAT.INLNCT.DIAG)	00003
CF3=FUCAT (PELMAX)#FLOAT (INLNCT)/FLOAT (TCDEL)	00004
CF 4=FLOAT (PELMAX) +FLOAT (INLNCT) /FLOAT (TCDATA)	00004 _00004
WRITE(6,2030) CF3,CF4	00004
2030 FORMAT (OC) MPRESSION FACTOR FOR G3 MACHINE (CF3) = .F8.4/	00004
C ** *********************************	-0000 ♦
CALL ERRY ES (PEL BUF. OTBUF. PEL MAX. VRES. ERRCNT)	00004
	70004
STOP	00004

	EN C SUBRCUTINE GETLG(LBITS.MODE.WRD.L)	0000
	IMPLICIT INTEGER(A-Z)	0000
****	++ LABELEC COMMON /G328IT/ ++++++	0000
	COMMON /GJERIT/MASK(JE)vCOMASK(JE)vLIDIT(JE)vLZGIT(JE)	-0000
	INTEGER AASK, COMASK, LIBIT, LZBIT	0000
		0000
	COMMEN/BJ FF/PELBUF(60,2),CDBUF(240),	0000
•	OTBUF(60.2).STFBUF(240). STAT(3))). COMMON/HLFF/CODE(3.92.2).CDDS(3.68.6).PREDCI(16).NPRED(16).	0000
•	CTABLE(16), CSTART(16), STBUF(1728), STRUN(1728)	0000
	COMMON /ER AY /ERROR S(2500)	0000
****	• • • • • • • • • • • • • • • • • • •	-0000
	COMMENSATIVE AD ADEL MAY LUCKE COLLAGE CHOMAN EDDWING LANDAN M	0000
	COMMON/IV AR/PELMAX.VRES.EPHASE.CMPMAX.ERRMUD.LINMAX.K COMMON/PVAR/INLNND.DYLNNC.DYELW.INELP.CDFLP.CTELP.CDELW.	0000
1	CDELCT. INELCT. TCDATA. TCDEL . ERRPNT. ERRUFF. ERRLIM.	0000
	ERRCHT & INLACT & CONSEC & I NNOBE & ZCNT & WROBUF & PACK &	0000
4	INCOD INREF, CTCOD, CTREF, TSTEBT	0000
	CCMMON/ICHAR/DD-II-MM.TT-NN-YY	0000
	COMMEN/LOGIC/SEARCH V DI AGVSYNC V LSS , WRITE V CHCOL V UNE LOGICAL SEARCH, DIAG, SYNC , LSS , WRITE , CHCCL , CNE	-000 (
****	**************************************	
		0000
	MCCE=4	0000
	TO IFUE NEW DIE COOK	-000
RE	TRIEVE NEXT BIT FROM COBUF	0000
-00	CONTINUE	0000
		0000
<u> </u>	ICODE A NEW LINE IF NECESSARY	000
		0000
	IF(L+CDELP.LE.CDELCT) GD TD 200 IF(CDELCT-C)ELP+1) 170.190.180	0000
170	STOP 170	0000
	CONTINUE	0000
	STFBUF(1) = 1 48(STFBUFVCDELPVCDELET-CDELP+1)	-0000
190	CONT INUE	0000
	COELF=32-(CDELCT-CDELP)	0000
00	CALL ENCIDI	0000
	POLEJAH(STERUE-COELP+L-1)	0000
	L=L+1	0000
	IF (PCL)220,300,240	0000
40	CONT INUE	- 0000
. 40	IF(ZCN T-1 3)310,260,340	0000
60	ZCNT=0	0 000
	GO TO 100	000
100	ZCNT=ZCNT+1	0000
	IF (LSS) 30 TO 380 GO TO 320	0000
310	CONTINUE	-0000
•••	ZCNT=0	000
20	CONT INUS	0000
	LPACK=LPACK+1	0000
324	IF (POL) 324,330,325 STCP 324	0000
	CONTINUE	000
	CALL MIZE (>OL. WRDBUF, LPACK, 1)	0000
350	- CENTINUE	-000
•	IF (LPACK-LT-LBITS) GO TO 100	0000
	WRC= 14 B(#ROBUF, 1,LPACK) MOCE=1	000
	MODE 31 RETURN	0000
40	CCAT INUE	-000
	IF (LSS) GO TO 360	0000
	LSS= .T RUE.	0000
60	90 TC 100 MO CE= 3	- 000 0
	RETURN	0000
150	CONTINCE	000
	MO CE=2	000
	<u>GETUAN</u>	000
	END SUBSCOUTING ENCODE	0000
	SUBROUTINE ENCODE	0000
	IMPL IC IT INTEGER (4-2)	0000
		0000
***	** LABELED COMMON /G32BIT/ ******	0000
		0000

INTEGER MASK.COMASK.LIBIT.LIBIT.COMPT(240).	00004: 00004: 00004: 00004: 00004: 00005: 00005: 00005: 00005: 00005: 00005: 00005:
COMMON/PUFF/PELBUF(60.2).CDBUF(200). OTBUF(60.2).STFBUF(200). COMMON/PERVY.CDSC1922.STFBUF(200). COMMON/PERVY.CRORS(2500) COMMON/PERVY.CRORS(2500) COMMON/PERVY.CRORS(2500) COMMON/PERVY.CRORS(2500) COMMON/PERVY.CRORS(2500) COMMON/PERVY.CRORS(2500) COMMON/ILES/TERMALPEIL.PEILITIONS COMMON/ILES/TERMALPEIL.PEILITIONS COMMON/IV.AR/PELBAX.VRES.EPHASE,CMPMAX.CRRMDD.LI.NMAN.K. COMMON/IV.AR/PELBAX.VRES.EPHASE,CMPMAX.CRRMDD.LI.NMAN.K. COMMON/IV.AR/PELBAX.VRES.EPHASE,CMPMAX.CRRMDD.LI.NMAN.K. COMMON/IC.HAR/DD.11.NM.TI.NN.TY COMMON/IC.HAR/DD.11.NM.TI.ND.TL.ND.TO COMMON/IC.HAR/DD.11.NM.TI.ND.TL.ND.TO COMMON/IC.HAR/DD.11.NM.TI.ND.TL.ND.TO INTICAL.T.1.PELBUF(1.INC.D).1=1.60) OTHER COMMON/IC.HAR/DD.TL.ND.TL.ND.TO INTICAL.T.1.ND.TL.ND.TL.ND.TL.ND.TL.ND.TO IF.(INLND.T.L.ND.TL.	000045 000045 000045 000045 000056 000056 000056 000056 000056 000056 000056 000056 000056
COMMON/IC HAS INDICATED TO TRUE (CO. 2). STREW (200). STAT(3000) **COMMON/IC HAS INDICATED TO TRUE (CO. 2). STATED (CO.	000049 000049 000049 000050 000050 000050 000050 000050 000050 000050 000050 000050
OTBUF(60.2).STRBUF(20). STAT(3000) COMMENTATIF, 400E(10).CSTART(10).STBUF(1723).STRUN(1728) COMMENTATIF, 400E(10).CSTART(10).STBUF(1723).STRUN(1728) COMMENTERAY ZERORS (2500) COMMENTERAY ZERORS (2500) COMMENTER STATEMAL DELLED COMMON VARIABLES COMMON/IV AR /PELMAX.VRES.EPHASE.CMPMAX.ERRMOD.LINMAX.K COMMON/IV AR /PELMAX.VRES.EPHASE.CMPMAX.ERRMOD.LINMAX.B COMMON/IV AR /PELMAX.VRES.EPHASE.CMPMAX.ERRMOD.LINMAX.B COMMON/IV AR /PELMAX.VRES.EPHASE.CMPMAX.ERRMOD.LINMAX.B COMMON/IV AR /PELMAX.B COMMON/IV AR /	000045 000045 000050 000050 000050 000050 000050 000050 000050 000050 000050 000050 000050 000050
COMMON/IC HAR INCOME STATE (16) COSTANT(16) STRUM(1728) STRUM(1728) COMMON/ELES/TERM+LDEIL-DELEIL-DIELL-ERFIL COMMON/IV AR /PELMAX, VRES.EPHASE, CMPMAX, ERRMOD, LINAA, K COMMON/IV AR /PELMAX, VRES.EPHASE, CMPMAX, ERRMOD, LINAB, K COMMON/IC HAR INCOMINATE, CONSC.LINAOSE, 2CXI, WADBUF, LPACK, O COMMON/IC HAR INCOMINATE, CONSC.LINAOSE, WATER CHECKLONGE COMMON/IC HAR INCOMINATE, CONSC.LINAOSE, CONSC.LIN	000045 000050 000050 000050 000050 000050 000050 000050 000050 000050 000050 000050
COMMCN/ERAY/ERRORS(2500) COMMCN/FILES/TERM.LPEIL.PEIL.DIFIL.PREIL COMMCN/FILES/TERM.LPEIL.PEIL.PEIL.DIFIL.PREIL COMMCN/FILES/TERM.LPEIL.PEIL.PEIL.PEIL.PEIL.DIFIL.PREIL COMMCN/FV AR /FL.MAX.VRES.EPMASE.CHPMAX.ERRMOD.LINAR.K COMMCN/FV AR /IN.MAY.OTLINNO.OTEL.W.INELP.COEL.DITLP.COE.W. ERRCHT, INLNO.TCOMSC.LINNOBF.ERRUPFILERRUFF.ERRLIN. ERRCHT, INLNO.TCOMSC.LINNOBF.ERRUPFILERRUFF.ERRLIN. COMMON/ICHAR/OD.II.MM.IT.NO.YV COMMCN/LOGIC/SEARCH.DIAG.SYNC.LSS.WRITES.CHDONE COMMCN/LOGIC/SEARCH.DIAG.SYNC.LSS.WRITES.CHDONE INTEGR INDEX(16) COMMCN/LOGIC/SEARCH.DIAG.SYNC.LSS.WRITES.CHDONE COMMCN/LOGIC/SEARCH.DIAG.SYNC.LSS.WRITES.CHDONE COMMCN/LOGIC/SEARCH.DIAG.SYNC.LSS.WRITES.CHDONE COMMCN/LOGIC/SEARCH.DIAG.SYNC.LSS.WRITES.CHDONE COMMCN/LOGIC/SEARCH.DIAG.SYNC.LSS.WRITES.CHDONE COMMCN/LOGIC/SEARCH.DIAG.SYNC.LSS.WRITES.CHDONE COMMCN/LOGIC/SEARCH.DIAG.SYNC.LSS.WRITES.CHDONE COMMCN/LOGIC/SEARCH.DIAG.SYNC.LSS.WRITES.CHDONE CODELCT-32 COMMCN/LOGIC/SEARCH.DIAG.SYNC.LSS.WRITES.CHDONE COMMCN/L	000045 00005 00005 00005 00005 00005 00005 00005 00005 00005 00005 00005
CCMMCM/FILES/TERM.LPEIL.PEIEIL.DIFIL.ERFIL. CCMMCM/FILES/TERM.LPEIL.PEIEIL.DIFIL.ERFIL. CCMMCM/FILES/TERM.LPEIL.PEIEIL.DIFIL.ERFIL. CCMMCM/PVAR/TML.NM3.DIL.NNG.DIEL.W.INDELP.COELP.JTALP.CDS.W. COMMCM/PVAR/TML.NM3.DIL.NNG.DIEL.W.INDELP.COELP.JTALP.CDS.W. CDELCT.INELCT.TCDATA_TCDELCT.W.W.JBUF.LPACK. COMMCM/PVAR/TML.NM3.DIL.NNG.FILED.LOELP.JTALP.CDS.W. COMMCM.TCHAR/DD.II.MM.TT.NN.VY COMMCM.TCHAR/DD.II.MM.TCHAR/DD.	000050 000050 000050 000050 000050 000050 000050 000050 000050 000050 000050 000050
CCMMCM/FILES/TERW.LPFIL.DFIL.DTFIL.FRFIL C***********************************	00005 00005 00005 00005 00005 00005 00005 00005 00005 00005 00005 00005
COMMON/I LESCIERNIDELLED COMMON VARIABLES COMMON/IVAR/PELMAX, VARES.EPHASE, CMPMAX.ERRMOD.LINMAX.K COMMON/IVAR/PELMAX.Y. NRES.EPHASE, CMPMAX.ERRMOD.LINMAX.K COMMON/IVAR/PELMAX.Y. NRES.EPHASE CMPMAX.ERRMOD.LINMAX.K COMMON/IVAR/PELMAX.Y. NRES.EPHASE.CMPMAX.ERRMOD.LINMAX.K ERRONT. INLOCT.CONSEC.LINMOS.F. ZENT. WRUBUF.EPACK. COMMON/ICHARDOD.LINMEF.DY.COD.OTREF.ISTRIT COMMON/ICHARDOD.LINM.TINN.YY COMMON/ICHARDOS.LINM.TINN.YY COMMON/ICHARDOS.LINM.TINN.TINN.TINN.TINN.TINN.TINN.TINN.T	000050 000050 000050 000050 000050 000050 000050 000050 000050 000050
COMMON/IVAR/PELMAX, WRES.EPHASE.CMPMAX.ERRMOD.LINMAM.K COMMON/PYAR/INLAND.OTTLAND.OTELWINELP.COELP.STEUP.CO.W. COMMON/PYAR/INLAND.OTTLAND.OTELWINELP.COELP.STEUP.CO.W. COMMON/PYAR/INLAND.OTTLAND.OTELWINELP.COELP.STEUP.CO.W. COMMON/PYAR/INLAND.OTTLAND.OTELWINELP.COELP.STEUP.CO.W. INCOLDINGE.CHONGE.CLANDOSF.ZCAT.WRUBUF.LPACK. INCOLDINGE.CHONGE.CLANDOSF.ZCAT.WRUBUF.LPACK. COMMON/ICHAR/DOSII.MM.TIANNYY COMMON/ICHAR/DOSII.MM.TIANNY COMMON/ICHAR/DOSII.MM.TIANNYY COMMON/ICHAR/DOSII.MM.TIANNY CO	000050 000050 000050 000050 000050 00005 00005 00005 00005 00005
COMMON/IVAR/PELMAX, WES.EPHASE, CMPMAX, ERRIMOD, LINMAR, K COMMON/PYAR/INLMNJ.OTLNNO.OTELWS, INELP, COELP, JTELP, COELWS COMMON/IC HAR/DOJ.INBET, OTCONOCC, LNNOBE, ZCNT, WROBUF, LPACK, OERCHY, INCOD.INBET, SITTED COMMON/IC HAR/DOJ.I, MM, ITANN, YY COMMON/IC HAR/DOJ.IC HAR/D	00005 00005 00005 00005 00005 00005 00005 00005 00005 00005
COMMON/IVAR/PELMAX, WRESSEPHASE, CMPMAX, ERR MOD. LI NMAX, K COMMCN/PVAR/IM, NND. OT LND. OT ELW, INCLP, CDELP, JTLLP, CDELP, BRILLIM, OELCT, TRECTT, INLACT, CONSEC, LNNOBE, ZCNT, WRUBUF, LPACK, INCODAINEE, DYCODAOTREE, TSTERT COMMON/ICHAR/DOLII, MM. TINN, YY COMMON/ICHAR/DOLII, MM. TINN, TINN, TINN, TINN, TINN, TINN, TINN, TINN	00005 00005 00005 00005 00005 00005 00005 00005 00005
COMMCN/PY AR/IN.NN.3.0TLNNO.OTELW.INELP.CDELP.JTLLP.CDELW. # COMMONT.INERT.GONEC.LNNOBF.ZCNT.WRUBUF.ERRLIN. # COMMONT.INERT.GONEC.LNNOBF.ZCNT.WRUBUF.ERRLIN. COMMONT.ICHAR/DO.II.MM.TI.NN.YY COMMONT.ICHAR/DO.II.MM.TI.NN.YY COMMONT.ICHAR/DO.II.MM.TI.NN.YY COMMONT.ICHAR/DO.II.MM.TI.NN.YY COMMONT.ICHAR/DO.II.MM.TI.NN.YY COMMONT.ICHAR/DO.II.MM.TI.NN.YY COMMONT.ICHAR/DO.II.MM.TI.NN.YY COMMONT.ICHAR/DO.II.MM.TI.NN.YY COMMONT.ICHAR/DO.II.MM.TI.NN.YY COMMONT.ICHAR.ICH	00005(00005(00005) 00005 00005 00005 00005 00005 00005 00005
COMMON/ICHAR/DD. II, MM. TI, TINN, YY COMMON/ICHAR/DD. II, MM. TI, NN, YY COMMON/ICHAR/DD. II, NN, YY COMMON/ICHAR/DD. I	00005 00005 00005 00005 00005 00005 00005 00005
# ERRCAT. INL.NCT.CONSEC. LENDOBF.ZCAT. WRUBUF.LPACK. INCON_INBEF.DTCID_TREF.ISTERT COMMON/IC HAR/DD.II.,MM.TT.NN.YY COMMCN/LOGIC/SEARCH.DIAG.SYNC.LSS. WRITE.CHCJDNE LOGICAL SEARCH.DIAG.SYNC.LSS. WRITE.CHCJDNE INTEGER INDEX(16) C INITIALIZE VARIABLES C CDELCT=32 CDELCT=32 CDELCT=32 CDELCT=32 CDEUTINE C CONTINUE C READ INPUT PICTURE FILE C CONTINUE C READ INPUT PICTURE FILE C CONTINUE IF (MODICINLAND.100).Eq.() WRITE(6:10) INLNND 110 FORMAT(' INPUT LINE NO. = '.16) IF (MODICINLAND.100).Eq.() WRITE(6:10) INLNND 110 FORMAT(' INPUT LINE NO. = '.16) IF (INELCT.LT.PELVAX) CALL EXIT C LCAD OUTPUT LINE NUMBER BUFFER C LCAD OUTPUT LINE NUMBER BUFFER C LOCATINUE C LOCATINUE C LOCAL CODE (6.7.T.CDELCT.CDDATA) 130 CONTINUE O L30 I=1.6 TAD C CALL CODE (6.7.T.CDELCT.CDDATA) 131 CONTINUE O L35 I=1.5 STERDUC(I) = COBUPC(I)	00005 00005 00005 00005 00005 00005 00005 00005
INCODA INBRES OT CONSTRET STERT COMMON/IC MAR/DDD.II,MM.TI.NN.YS COMMON/IC MAR/DDD.II,MM.TI.NN.YS COMMON/IC MAR/DDD.II,MM.TI.NN.YS COMMON/IC MAR/DDD.II,MM.TI.NN.YS COMMON/IC MAR/DDD.II,MM.TI.NN.YS COMMON/IC MARKET COMMON	00005 00005 00005 00005 00005 00005 00005 00005
COMMON/IC HAR/DD.11, MM.TT.NN.YY COMMON/IC GIC/SERCH, DIAG.SYNC.LSS, WRITE.CHC).DNE LSGICAL JEARCH.D1AG.SYNC.LSG.WRITE.CHCOLONE INTEGER INDEX(16) C CINITIALIZE VARIABLES C CDELCT=32 CDEATA-8 DC 50 I=2.240 CDBUF(1)=0 STFBUF(1)=0 C READ INPUT PICTURE FILE C CREAD INPUT PICTURE FILE CREAD INCOMMONIONINGLET.OFELBUF(1.INCCD).1=1.60) IF (MODININGLET.CF.PELBUF(1.INCCD).1=1.60) IF (MODININGLET.CF.PELBUF(1.INCCD).1=1.60) IF (MODININGLET.CF.PELBUF(1.INCCD).1=1.60) IF (MODININGLET.LT.PELMAX) CALL EXIT CLAD OUTDUT LINE NOW =*'.150 CLAD OUTDUT LINE NUMBER BUFFER CLANCEF=INLIND IF (SEARCH.) TINNOBF CLANCEF=INLIND IF (SEARCH.D) TINNOBF CLANCEF=INLIND IF (INLINCT.ST.O) STOP DD 130 I=1.6 CALL CODEG(67.T.CDELCT.CDDATA) 130 CONT INUE OCALL CODEG(67.T.CDELCT.CDDATA) OOLIGI = 1.56 STFBUF(1) = COBUF(1)	00005 00005 00005 00005 00005 00005 00005
LOGICAL SEARCH-DIAG-SYNC-LESG-WRITE-CHCOL-ONE INTEGER INDEX(16) C	00005 00005 00005 00005 00005 00005 00005
INTEGER INDEX(16) C C C C C C INITIALITE VARIABLES C C CDELCT=32 CDATA-0 CC SO 1=2.240 CDBUF(1)=0 STFBUF(1)=0 STF	00005; 00005; 00005; 00005; 00005;
C INITIALIZE VARIABLES C CDELCT=32	00005 00005 00005 00005 00005
C INITIALIZE VARIABLES C CDELCT=32 CDEATA=0 CC 50 I=2,240 CDBUF(I)=0 ST FBUF(I)=0 50 CONTINUE C READ INPUT PICTURE FILE C CONTINUE REAC(1,E)C=120,ERR=500) IF (MOD(INLNO,100),EQ.0) WRITE(6,110) INLNO IF (MOD(INLNO,100),EQ.0) WRITE(6,110) INLNO IF (IND(INLNO,1-1,425),NE_0) GO IC 100 IF (INCCT+1,41,NC) TO 1,40,NC) C LCAD OLTPUT LINE NUMBER BUFFER C LNCCEF=INLNO IF (SEARCH) J TLNNO=LNNOBF C WRITE SIX LOLIS C ISO CONTINUE C ISO	00005: 00005: 00005: 00005:
C INITIALIZ VARIABLES C CDELCT=32 CDBATA=0 CC 50 I=2.240 CDBUF(1)=0 STFBUF(I)=0 STFBUF(I)=0 C READ INPUT PICTURE FILE C 100 CONTINUE REAC(1.5ND=120.5RR=500) * INLNNO,I NELCT.6PELBUF(I.1NCCD).I=1.60) IF (MODI(INLNNO,1D01.6Q.0) WRITE(6.110) INLNNO IF (MODI(INLNNO-1.492E.3).MR=0) CD IC 100 IF (INCLOT.1.4.0PELWAX) CALL EXIT C LCAD OUTPUT LINE NUMBER BUFFER C LNNCEF=IJALNNO IF (SEARCH) DILNNOSHNNOBF C IF (INLNNJ.E.LINMAX) GD TO 140 C WRITE 31X COL1*3 C 120 CONTINUE C IF (INLNJ.E.LINMAX) GD TO 140 C WRITE 31X COL1*3 C CALL CDSE(67.T.CDELCT.CDDATA) 130 CONTINUE O 139 I=1.6 TFOUR (I) = CDBUF(I)	00005. 00005. 00005.
C INITIALIZ	00005 00005 00005
C CDELCT=32 CDEATA=0 DC 50 1=2.240 CDBUF(1)=0 STFBUF(1)=0 STFBUF(1)=0 C READ INPUT PICTURE FILE C REAC(1.5:4D=120.5RR=500) * INLNNO.INELCT.(PELBUF(1.INCCO).1=1.60) IF (MOD(INLNO.100).EQ.0) WRITE(6.110) INLNNO IF (MOD(INLNO.100).EQ.0) WRITE(6.110) INLNNO IF (MOD(INLNO.1-1.42E.3).MF.0) GO IC 100 C LGAD OLTPUT LINE NUMBER BUFFER C LGAD OLTPUT LINE NUMBER BUFFER C LGAD OLTPUT LINE NUMBER BUFFER C LGAD CONTINUE C TF (INLNO.LE.LINMAX) GO TO 140 C WRITE 31X EGL1*3 C CALL CODEG(67.T.CDELCT.CDDATA) 130 CONTINUE STFBUF(1)=CDBUF(1)	000051
C C SO I=2,240 CDBUF(1)=0 STFBUF(1)=0 STFBUF(1)=0 C C READ INPUT PICTURE FILE C REAC (1,2:40=120, ERR=500) * INLNNOSINELCT, (PELBUF(1, INCCD), I=1,60) IF (MOD (INLNNO, 100), EQ.0) WRITE(6,110) INLNNO 110 FOFMAT (' INPUT LINE NO, =',16) IF (IND(INLNNO-1,375), NF.0) GO IC LOO IF (INCCT, LT, PELMAX) CALL EXIT C LCAD OLTPUT LINE NUMBER BUFFER C LCAD OLTPUT LINE NUMBER BUFFER C LNNCEF=I,ILNNO IF (SEARCH) OLNNOSENNOBF C IF (INLNNO-1,215) STOP C C WRITE 31X EQ.1*3 C CALL COSG (67,T,CDELCT,CDDATA) 130 CONTINUE STEDUF(1)=CDBUF(1)	
CC 50 I=2,240 CDBUF(I)=0 ST FBUF(I)=0 C CONTINUE C READ INPUT PICTURE FILE (C READ INPUT PICTURE FILE (D CONTINUE REAC(I,END=120,ERR=500) **INLNNO,INELCT,(PELBUF(I,INCCO),I=1,60) IF (MCD (INLNNO,1001,EQ.0) WRITE(6,110) INLNNO IF (MCD (INLNNO,1142ES),NE-0),GO IC LOO IF (INELCT,LT,PELMAX) CALL EXIT (C LGAD OUTPUT LINE NUMBER BUFFER (C LGAD OUTPUT LINE NUMBER BUFFER (C LNCEF=INLNNO IF (SEARCH)) TLNNO=LNNOBF (C VRITE SIX EQLIS (C VRITE SIX EQLIS (C LGO CONTINUE (C LGO CO	
COBUF(I)=0 STFBUF(I)=0 CONTINUE C READ INPUT PICTURE FILE C REAC (I PICTUR	
STRUCTION CONTINUE	00005i 00005.
50 CONTINUE C READ INPUT PICTURE FILE 100 CONTINUE REAC(1, E: 10=120, ERR=500) * INLNNIO. INELCT. (PELBUF(1, INCCD). I=1.60) IF (MOD(INLNIO. 100). EQ.0.) WRITE(6, 110) INLNNU 110 FORMAT (' INPUT LINE NO. = '.16) IF (MOD(INLNIO. 1-1.4785).NE.0) GD IC 100 ON IF (MOD(INLNIO. 1-1.4785).NE.0) GD IC 100 C LGAD OLTPUT LINE NUMBER BUFFER C LNNCEF=INLNNO IF (SEARCH) DILNNO=LNNOBF C WRITE SIX EQLIS C 120 CONTINUE C 1F (INLNI). LE. LINMAX) GD TO 140 C WRITE SIX EQLIS C 120 CONTINUE C CALL CDEG(67.T.CDELCT.CDDATA) 130 CONTINUE STFBUF(I) = COBUF(I)	00005.
C READ INPUT PICTURE FILE C 100 CONT INUE	00005
C	00005
100 CONTINUE	00005.
# AC (1 *** NC=120 ** ERR=500) * INLNNO. INELCT ** (PSEBUF(1 ** INCCD) ** I=1 ** 60) IF (MCD (INLNO. 100) ** 60 ** 0 WRITE(6 ** I10) INLNNO 110 FOFMAT (' INPUT LINE NO. = ' . 16) IF (MCD (INLNNO-1 ** V3ES) ** NE ** 0) GO IC 100 IF (INELCT ** LT ** PEL YAX) CALL EXIT C LCAD GLTPUT LINE NUMBER BUFFER C LNRCEF ** INLNNO IF (SEARCH) O TLNNOSF C WRITE SIX LOLIS C 120 CONTINUS C 17 (INLNO. STOP C 17 (INLNO. STOP C 17 (INLNCT. ST. 0) STOP C 18 (19 CODEG(67 ** T. ** CODELCT. ** CODATA) 10 CONTINUS STFBUF(1) = COBUF(1)	00005:
# INLNNO.INELCT.(PELBUF(I.INCCD).I=1.60) IF (MOD(INLNNO.IOO).EG.O) WRITE(6.IO) INLNNO 110 FOFMAT(' INPUT LINE NO. = '.16) IF (MOD(INLNNO-1.VZES).NE.O) GO IC 100 IF (INELCT.LT.PELMAX) CALL EXIT ONLINE INLNCT = INLNCT+1 C LCAD OLTPUT LINE NUMBER BUFFER C LNKCEF=INLNNO IF (SEARCH) OTLNNO#F OF THE SIX EQLIPS C 120 CONTINUE C IF (INLNNJ.LE.LINMAX) GO TO 140 OF THE SIX EQLIPS C 120 CONTINUE C IF (INLNCT.GT.O) STOP OF THE SIX EQLIPS C ALL CODEG(67.T.CDELCT.CDDATA) 130 CONTINUE C STFBUF(I) = COBUF(I)	99995 1
IF (MCD (INLNNO, 100) = 0.0) WRITE(6, 110) INLNNO 110 FORMAT (* INPUT LINE NO. = *.16) O IF (MCD (INLNNO-1.4XES) NE.0) GO IC LOO IF (INELCT .LT.PELMAX) CALL EXIT O INLNCT=INLNCT+1 O C LGAD OLTPUT LINE NUMBER BUFFER O LNKCEF=INLNNO O IF (SEARCH) OTLNNO=LNNOBF O C URITE SIX EQLI*S O C VRITE SIX EQLI*S O C C C C C C C C C	00005. 00005.
110 FORMAT (' INPUT LINE NO. = '.16) IF (MOD(INLNNO-1.4785).NE-0) GO TC 100 IF (INELCT .LT.PELMAX) CALL EXIT OINLNCT=INLNCT+1 C LGAD OLTPUT LINE NUMBER BUFFER C LNNCEF=INLNNO IF (SEARCH) OTLNNO=LNNOBF OIF (INLNNJ.LE.LINMAX) GO TO 140 C WRITE SIX LOL1*S C 120 CONT INUS C IF (INLNCT.GT.O) STOP CO 130 I= 1.6 I= 0 CALL CODEG(67.T.CDELCT.CDDATA) 130 CONT INUS STFBUF (I) = COBUF(I)	00005.
IF (MOD(INLNNO-1.VRES).NF.O) GO TO 100 IF (INELCT .LT.PELMAX) CALL EXIT OUT INLNCT=I NLNCT+1 OUT INLNCT=I NLNCT+1 OUT INLNCT=I NLNCT+1 OUT INLNCT=I NLNCT+1 OUT INLNCEF=INLNNO OUT INLNCEF=INLNNO OUT IT (SEARCH) OUT INNO=LNNOBF OUT INNO=LNNOBF OUT INNO=LNNOBF OUT INNO=LNNOBF OUT INNO=LNNOBF OUT INNO=LNNOBF OUT INDEED OUT	00005.
INLNCT=INLNCT+1 C LGAD OLTPUT LINE NUMBER BUFFER C LNKCEF=INLNND IF (SEARCH) OTLNNO=LNNOBF OF IF (INLNNJ-LE-LINMAX) GO TO 140 C WRITE SIX EGL1-S C 120 CONT INUE C IF (INLNCT-GT-O) STOP CO 130 I=1.6 I=0 CALL CODEG(67.T.CDELCT.CDDATA) 130 CONT INUE STFBUF(I)=COBUF(I)	00005
C LCAD OLTPUT LINE NUMBER BUFFER C LNKCEF = IJLINNO	00005.
C	00005.
C	00005 .
LNKCBF=INLNNO	00005. 00005.
IF (SEARCH) 3 TLNNO#LNNOBF IF (INLNNJ.LE.LINMAX) GO TO 140 C WRITE SIX EQL1*S 120 CONT INUE IF (INLNCT.GT.0) STOP CO 130 I=1.6 I=0 CALL CODEG(67.T.CDELCT.CDDATA) 130 CONT INUE STFBUF(I) = CDBUF(I)	00005
IF (INLNNJ.LE.LINMAX) GO TO 140 C WRITE SIX LOLI'S C 120 CONT INUE C IF (INLNCT.GT.O) STOP CO 130 I=1.6 I=0 CALL CODEG(67.T.CDELCT.CDDATA) 130 CONT INUE STFBUF(I)=CDBUF(I)	00005.
C WRITE SIX EQLI'S 120 CONT INUE 0 IF (INLNCT.GT.0) STOP CO 130 I=1.6	00005
C 120 CONT INUE 120 CONT INUE C IF (INLNCT.GT.0) STOP CO 130 I=1.6 T=0 CALL CODEG(67.T.CDELCT.CDDATA) 130 CONT INUE 0 135 I=1.6 STFBUF(I)=COBUF(I)	00005
C 120 CONT INUE 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	00005. 00005 .
120 CONT INUE C	00005
C	00005
T=0 CALL CODEG(67.T.CDELCT.CDDATA) 130 CONT INUE B9 135 i=1;5 STFBUF(I)=CDBUF(I) 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	00005
CALL CODEG(67.T.CDELCT.CDDATA) 130 CONTINUE 00 00 135 i=1.5 STFBUF(I) = CDBUF(I) 0	00005
130 CONTINUE 0 09 135 1=1+5 STEBUF(I)=CDBUF(I)	00005
00 135 1=1v5 STFBUF(I)=CDBUF(I)	00005
STFBUF(I)=CDBUF(I)	00005: 00005
	00005:
	00005
GD TD 400	00005
C FIRST CF K LINES?	00005:
	00005
	00005
## 1F (MOD (INLNCT - 1 + K) + NE+ 0) GO FO 600	00005: 00005 -
c .	00005
C OHE-DIMENSIONAL CODING O	00005
WRITE ONE SOLI	
	000051
	000051
CALL CODESTORTICUELCTICUDATA!	00005: 00005:
	00005: 00005: 00005:
C	00005t 00005t 00005t 00005t
C TEST COLOR OF FIRST ELEMENT 0	000051 000051 000051 000051
The state of the s	00005t 00005t 00005t 00005t
1f(14f(Palbuf(1.1HCOD).1.1).EQ.0) GO TO 150 0	00005: 00005: 00005: 00005: 00005: 00005:

FIRST ELEMENT BLACK! ENCODE O-LENGTH WHITE RUN	00005
CALL GCOOLR(0, 1, CDELCT, CDDATA)	00005
POLAR=2	00005
CALCULATE RUN LENGTH AND ENCODE	00005
150 CONTINUE	00005
RU N=0	00005
DO 200 I=1.9ELMAX PEL=148(>EL BUF(1, INCO)), I, 1)+1	00005
IF(PEL.ÉQ.POLAR) GO TO 180	00005
GALL GCOSLA(RUN) POLAR (COELCT (COOATA) IF (.NOT.) IAG) GO TO 170	
WRITE(6,160) RUN, POLAR, CDELCT, CDDATA	00005
160 FURNAY (418) 170 CONTINUE	00005
QUN=1	00005
POLAR= MOD (POL AR+2, 2)+1 GO TC 200	000 <i>0</i> 5 000 <i>0</i> 5
-100-CONT-BNUC	
RUN=RUN+1 200 CONTINUE	00005 00005
CALL GUDLERGRUN, POLAR, CDELCT, CDDAYA)	00005
IF(.NOT.)143) G) TO 210 WRITE(6.160) RUN.POLAR.CDELCT.CDDATA	
GO TO 210	00005
Pub Gruera Lauri Aggrica	00006
TWO-DIMENSIONAL CODING	00006
600 CONTINUE	00006
WRITE CHE ECL2	00005
T=0 CALL CODEG(68,T,CDELCT,CDDATA)	00006 00006
INITIALIZE ARRAY POINTERS	00006 00006
J=!	00006
DD 610 [=1.16 INCEX(I)=0	00006
61 0 CONTINUE	00006
00 700 I=1.PELMAX	00006
PREDICT NEXT ELEMENT	00006
IF(I-1) 611.612.613	00006
611 STOP 611	00006
PEL 1=1 48(PEL BUF(1 + I NREF) + I + 2)	d0000 d0000
PEL2=0	00006
CALL M123 (PEL1 vPEL2 v32-2 v2) GD TO 615	00006
613 CONTINUE	00006
PEL 2=1 43(PEL BUF(1.1NCJD).1-1.11 PEL 148(PEL BUF(1.1NR 2F).1-1.3)	00006 00006
CÂLL 4123 (ĐẾLÍ - ĐẾL2 - 32-3 - 3)	00006
615 CONT INUE SPI=FEL2+1	00006 00006
JS=1NBEX(-S-1)	
1F(JS-1) 620,630.630	00006 00006
ADD A STATE ENTRY TO TEAL	00006
620 CONTINUE	00006
STEUF(J)=SPI	00006
STRUN(J)=1	00006
IN CEX (5° 1)= J JS*J	
Ĵ≠J+1	00006
630 CONTINUE IF (PREDCT(SP1).EQ. 148(PELBUF(1,INCDD). 1.1))GG TO 650	00006 00006
1NCEX(5)1x0	00006
GD TC 700 650 CONTINUE	00006 00006
\$7 FUN(J\$) = 3 TRUN(J\$) +1	
700 CONTINUE	00006 00006
CONSTRUCT CODE LINE	9 0 0 0 0
	00006

0-8

	00006
JMAXEJ-1 CO 900 J=1, JMAX	00006
	00006
CALL CODEG(STRUN(J).CTABLE(SP1).CDELCT.CDDATA)	00006
IF (.NOT. JIAG) GO TO 800	
WRITE(6.160) STRUN(J).S.CDELCT.CDDATA	00006
800 CONTINUS	00006
210 CONTINUE	00006
C SWITCH CODE & REFERENCE LINES	
C	00006
TEMP=I NREF	00006
INCOD=TE 4P	00006
C	000061
BIY STUFFING (INSERT ONES)	00008
C CALL STUFFI (CDBUF, STFBUF, STFBIT, CDELCT)	00006
C C	00006
C SAVE LINE LENGTH (DATA + EOL)	00006
STAT(I NLNCT)=CDDATA+CDCS(1.68.1)	00006
	. 00006
C CHECK CODED LINE LENGTH	00008
C FILL SC VOVIAX = LCDFL CT = 32)	00006
FILL=CM9NAX=(CDFLCI=32) IF (FILL) 400.400.250	00006
C	00006.
C CODE LINE FOO SWORTS FILL IT TO CHPMAX	00006
250 CONTINUE CDELCT =COELCT +FILL	00006
COLE CI -COLE CI VI LEC	30000
C ACCUMULATE STATISTICS AND ERROR CORRUPT	00006
400 CONTINUE	00006
IF(ERRAD).EQ.NN) GD. TO 390	00006
C ERROR CORRUPT	00006 00006
S50 CONTINUE	00006
FRRBIT=ERRORS(ERRPNT)-ERROFF-TCDEL	00000
IF (EERRIT 1 2.0) GO TO 360	00006
IF(ERRBIT.GT.CDELCT-32) GC TC 390	00006
C PROP IN RANGE OF CODED LINE; CHANGE APPROPRIATE DIT	
C attitude to the constraint in	00007 00007
err=148(5TF8UF, ERRBIT+32,1) BYT=MOD(3TY+1,2)	30007
CÂLL MIZA (BIT.STFBUF.ERRBIT+32.1)	00007
ERECNT *ERRCNT +1	00007
C INCREMENT ERROR LIST POINTER	00007
360 CONT INUE	00007 00007
ERRENT =ERRENT +1 IF (ERRENT -LE-ERRLIM) GO TO 350	00007
C	00007
C ERROE LIST EXHAUSTED	00007
C ER F F N T = EH RP N T - 1	00307
	00007
370 FORMAT (') ERROR LIST EXHAUSTED AT', IIC, 'TH =RRUR''/	00007
+ 'LAST ERROR OCCURRED AT'.110." HITS!)	00007
C	00007
	00007
C 390 CONTINUE 1	00007 00007
TCCATA=TCOATA+CDCATA	00007 00007
TSTEBT=TSTEBT+STEBIT IF(DIAG) WRITE(5,160) INLNCT, CDDATA	00007
C	00007
IF (NOT	10000
CDELW=(CJELCT+32-1)/32 WRITE(6,450) (CDBUF(I):I=1:CDELW)	00007 00007
WRITE(01430) (COSOF(1)11=14COECH)	00007
450 FCFMAT (6212)	0000:
WRITE(6,445) STERLT 445 FORMAT(13, ONES INSERTED)	00007
460 CONTINUE	00007
	· · ·

JNC_ASSIFIED

	RETURN	000073
C		000073
	CALL EXIT	000073
C		000074
<u> </u>		000074
_	SUBROUTINE CODEG(LEN.TABLE.CDELCT.CDDATA)	000074
<u>c</u>	IMPLICIT IN TEGER (A-Z)	000074
	COMMCN/BUFF/PELBJF(60,2), CDBUF(240),	000074
	0 TRUE (60.2) . STERUE (240) . STAT (3220)	000074
	COMMON/HJFF/CODE(3.92.2).CODS(3.68.6).PREDCT(16).NPRED(16).	000074
	CTABLE (16) .CSTART (16) .STBUF (1728) .STRUN (1728)	000074
	LCGICAL PREFIX	00007
C	•	000071
	18 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8	
C	Lalfn	00007:
	Tatané	00007
	PREFIX=•FALSE•	00007
-	IF(T) 1v600v3	00007
	1 STCP 1	00007
	S CONTINUE TF(L.G5.55.AND.T.LE.2) GO TO 100	00007
	IF (L.GE.33. AND. T.GE.3) GC TO 500	00007
	10 CONT INUE	00007
	CALL MI28(CODS(3.L.T).CDBUF.CDELCT+1.CCDS(1.L.T)) CDELCT=CDELCT+CDDS(1.L.T)	00007
	- If (b vit vob) RETURN	-00007
	CDOATA=CJDATA+CODS(1.L.T)	00007
	IF(.NOT.PREFIX) RETURN	00007
	CALL MIZE(LENGTH-CDBUF-CDELCT+1-11)	00007
	CDELCT=CDELCT+11CDCATA=CDCATA+11	00007
	RE TURN	00007
1	100 CONTINUE	00007
		-00007 00007
	LENGTH=L	00007
	L=cc	00007
	PREFIX=+TRUE+	00007
	GO IC 10	00007
	LENGTH=L	00007
	Lace	00007
	PREFIX=.TRJE.	00007
	GO TO 10 IZO CONTINCE	00007
•	L#65	00007
	60 TC 10	_0000Z
:	500 CONTINUS LENGTH=L	00007 00007
		-00007
•	PRĒF IX=•TRUE•	00007
,	GO TC 10	00007
C	WRITE EOL	00007
غ	# 11 & 1 & Wh	00007
(SOO CONTINUE	00007
	T=1	00007
		- 00007 00007
	SUBROUTINE ONEGER(INDEX.COLOR.STATUS.L)	00007
	IMPLICIT INTEGER(A-Z)	00008
C#1	***** LABELLO COMMON /G32BIT/ ******	80000 80000_
	COMMON /J328IT/MAS<(32).CGMASK(32).LI8IT(32).LLBIT(J2)	00008
	INTEGER 4ASK.COMASK.LIBIT.LZBIT	0000a
c		-0000e
	COMMCN/BJFF/PEL9UF(60,2),CDBUF(240), # OTBUF(60,2),STFBUF(240), STAT(3333)	80000
	COMMON/HUFF/CODE(3,92,2),CDDS(3,68,6),PREDCT(16),NPRED(16),	8 C 0 0 0
	* CTABLE(16), CSTART(16), STBUF(1728), STRUN(1728)	00008
	COMMON /ELAY /ERROR S(2500)	20000
C#1	******************* FILE DEFINITIONS ************	80000 00008
	- COMMON/FILES/TERMVLPFILVPELFILVOTFILVERFIL	-00008
Ç		90006
<u>ç.</u>	**************************************	00003
C	COMMON/IVAR/PELMAX.VRES.EPHASE.CMPMAX.ERRMOD.LINMAX.K	00003 0000E
	WE THERE & THE THERE AND TANGETS THE TANGETS TO THE THERE AND THE	77700

UNCLASSIFIED

A STATE OF THE STA

	FRECHT I NU NCT CONSEC AL NINCEE SCHI AND DRUE AL PACK	80 000 80 000
•	INCOD.INREF.OTCOD.OTREF.TSTFBT COPMCN/ICHAR/OD.II.MM.TT.NN.YY	000082
	COMMONAL DELCASEARCH + DIAG + SYNC + LSS+ WRITE + CHCOL+ UNL	-000003i
_	LCGICAL SEARCH, DIAG. SYNC. LSS, WRITE, CHCOL. DNE	00008
Ca se s 1	**************************************	000082
C		000082
E BE	CIL DECOME A COOL DETOLEMENT MENT COOL CUCTUM	000082
C	GIN CECODE LOOP; RETRIEVE NEXT CODE WORD LENGTH (L)	000082
-1000-	CONT SIVE	-00000
	L=0 WRDBUF=0	00008
	LPACK=0	00008
1 002	LENBITECIDE (1.1 NDEX.COLOR)	00008
	CALL GET G(LENRIT.MODE.LBITS.L) IF(CIAG) WRITE(6:1003) LENBIT.MODE.LBITS.L	00008
1003	FORMAT (21 6, 28, 16)	00008
	40 70-(1+40+1200+1205+1490)+ HODE	-000000
1.04.0	STOP 1 04 0 CONT IN UE	000084
	IF (LBITS. EQ.CODE(J.INDEX.COLORI) CO TO 1100	000084
E N	• • • • • • • • • • • • • • • • • • • •	000084
<u>N</u>	MATCH: ADVANCE CODE WORD INDEX VIA DECODE THREAD	000084
	INDEX= CDDE(2. INDEX. COLOR)	000084
	If (1 NDDI v GC v 93) - GO - TO - 11 90	-00000
_	IF(CODE(1.INDEX.COLDR).EQ.LENBIT) GO TG 1040	000084
\(\) \(\) \(\	DE NORD LONGER: FROM THE TCP	-80000 -80008
č	The Edward Mark Title Top	00008
	6C 1C 1035	00008
C M/	TCH FOUND	00008
		-00000
1 100	CONT INUE	00008:
~	CDELP=CD=L3+L	80000 80000
Č N	T AN EQL	00008
C NO		-0000H
C 11	ST FOR MAKE UP OR TERAINATING CODE	90008
-	ST TON HALL OF ON TENTINATING CODE	-00003
	RUNLEN=INDEX-1	80000
	IF(INDEX.GE.65) RUNLEN=(INDEX-64)*64 IF(RUNLEN.EQ.0) GJ YD 1160	00008
	IF (CCL CR. EJ.I) GO TO 1155	00008
	IF (RUNLE LaL Tall) STOP 1100	00000
C A	D BLACK RUN TO DUTPUT BUFFER.	80000
	D SEACH NOT TO SOFFER	-00003
	DC 1150 I =1 .RUNLEN	8 0000
	CALL MI23(CCLOR-1.0TBUF(1.CTCCD).CTELP.1) CTELF=CTELP+1	00006 80000
	IF(CTEL9-1.GT.PELMAX) GO TC 1180	80000
1150	CONT INUE	80000
С	GC TC 1160	8 0000 B
		- 00005
C		80000
1155	CONTINUS CTELF=CTELP+RUNLEN	00005
	IF (OTELP-1.GT.PELMAX) GC TO 1180	00008
		00008
	PUT LINE LESS THAN OR EQUAL TO MAX SPECIFIED	8 0000
-1160	CONT INVE	00008
	IF(INDEX.LT.65) GO TO 1170	60000
	INCEX= 3	30000
c	GC YC 1030	00008
Č RI	IN ADDED TO DUTPUT INE: LENGTH LESS THAN OR EQUAL TO PELMAX (1)	00228
c	CANTINUE	60000
1 170	CONTINUE CHECKS STRUES	80000
•	STATUS=1	00000
	RETURN	00008
C RI	JN ADDED UNTIL PELMAX EXCEEDED: LINE TCC LONG (2)	00003
- *	or apply state please accepted time its time its	60000

1180 CONTINUE	000000
IELDIAGI WRITE(6.1185) (CTBUE(1.CTCCD).181.60)	000090
1185 FOFMAT (6210)	000090
STATUS=2	000090
RETURN	
c	000090
Č · NO MATCH FJUND IN CODE TABLE (3)	000090
	000090
1190 CONTINUE	000090
STATUSE3	000091
RETURN	000091
	000091
- EOL1 DET CCT CO (4)	000091
	000091
1500 CONTINGE	000091
ST ATUS =4	000091
RE TURN	000091
C EOL2 DETECTED (5)	000091
C College (3)	000092
1205 CONT INUE	
STATUS=5	000092
RETURN	000092
ENC	000092
SUBROUTINE TWOGER(INDEX.CCLCR.STATUS.L)	000092
INFLICIT INTEGER (A-Z)	000092
C****** LABELED COMMON /G328IT/ ******	000092
C	000092
COMMEN /332 CIT/MASK(32) COMASK(32) LIGHT(32) LEGIT(32)	00009 £
INTEGER MASK.COMASK.LIBIT.LZBIT	000091
<u>C</u>	000093
CUMMEN/BJFF/PELBUP(60, 21, CDBUF(240)	00009
* DTBUF(60,2),STF3UF(240), STAT(3000) CCMMCN/HUFF/CODE(3,92,2),CDDS(3,68,6),PREDCT(16),NPRED(16),	000093
* CTABLE(16),CSTART(16),STBUF(1728),STRUN(1728)	00009
CCPMCN/ERAY/ERRORS(2500)	000093
C+++++++++++++++++++++++++++++++++++++	
C .	000091
COMMCN/FILES/TERM.LPFIL.PELFIL.OTFIL.ERFIL	00009
2	00009
C+++++++++++++++++++++++++++++++++++++	*** 000094
C	00009
COMMON/IVAR/PELMAX, VRES.EPHASE.CMPMAX.ERRMOD.LINMAX.K	00009
COMMCN/PV AR/IN_NO.OTELW.DELP.COELP.JTELP.COELP.COELP.	000094
COELCTVINELCT VICOATA VECER PATVER ERRUFF VERRLIMV	
* ERRONT INLINCT CONSEC LANDBE ZONT WADBUF LPACK	000094
* INCOD. INREF. CTCDD. DTREF. TSTFBT	00009
COMMON /IC HAR /DD, II, MM, TT, NM, YY	00009
COMMCN/LJGIC/SEARCH, DIAG, SYNC, LSS, WRITE, CHCJL, JNE LOGICAL SEARCH, DIAG, SYNC, LSS, WRITE, CHCDL, CNE	00009
INTEGER 3 TONT (16)	00009
INTEGER STONE (10)	00009
Č++++++++++++++++++++++++++++++++++++	
C S S S S S S S S S S S S S S S S S S S	00009
ČINITIALIZE	00009
	00009
DO 100 I=1,16	00009
STCNT(1)=0	00006
100 CONTINUE	00009
C	00009
00 T000 3=1 yPCLM4X	00009
C	00009
C FIND STATE OF NEXT PREDICTED PEL	00009
	90000
IF (P-1) 611,612,613	00009
612 CONTINUE	00009
ATP # A11111AP	00009
PEL 1=1 AR(OTRUE(1-)TREE1-P-2)	
PEL1=148(078UF(1.07REF).P.2)	
PEL1=148(OT8UF(1.)TREF).P.2)	00009
PEL2-0	00009
CALL MI23 (PEL1 .PEL2 .32-2 .2)	
7512-0 CALL MI23 (PEL1 .PEL2 .32-2 .2) GO TO 615 613 CONTINUE PEL 2=1 43 (OTBUF(1.0 TCOD) .P-1 .1)	00009
PEL2=0 CALL MI23 (PEL1 .PEL2 .32-2 .2) GO TO 615 613 CONTINUE PEL2=1 43 (OTBUF(1.0 TCOD) .P-1 .1) PEL1=1AB(OTBUF(1.0 TREF) .P-1 .3)	00009 00009 00009 00009
CALL MI23 (PEL1 .PEL2 .32-2 .2) GO TO 615 613 CONTINUE PEL 2=1 43 (OTBUF(1.0 TCOD) .P-1 .1) PEL 1=1AS (OTBUF(1.0 TREF) .P-1 . 3) CALL MI23 (PEL1 .PEL2 .32-3 .3)	00009 00009 00009 00009 00009
CALL MI23 (PEL1 •PEL2 •32-2 •2) GO TO 615 613 CENTINUE PEL 2=1 43 () TBUF (1 •) TCO) • P-1 •1) PEL 1=148 () TBUF (1 •) TCO) • P-1 •1) CALL MI23 (PEL1 • PEL2 •32-3 •3) 615 CONT INUE	00009 00009 00009 00009 00009
CALL M123 (PEL1 .PEL2 .32-2 .2) GD TO 615 613 CONTINUE PEL2=1 43 (DTBUF(1.DTCOD) .P-1 .1) PEL1=1AB(.DTBUF(1.DTCOD) .P-1 .3) CALL M123 (PEL1 .PEL2 .32-3 .3) 615 CONTINUE 3P1=PEL2+1	00009 00009 00009 00009 00009
CALL M123 (PEL1 .PEL2 .32-2 .2) GD TO 615 613 CONTINUE PEL2=1 43 (DTBUF(1.DTCOD) .P-1 .1) PEL1=1AB(.DTBUF(1.DTREF) .P-1 . 3) CALL M123 (PEL1 .PEL2 .32-3 .3) 615 CONTINUE 371=PEL2+1	00009 00009 00009 00009 00009
CALL MI23 (PEL1 •PEL2 •32-2 •2) GO TO 615 613 CENTINUE PEL 2=1 48()TBUF(1 •) TCOD) •P-1 •1) PEL1=1A8()TBUF(1 •) TCOD) •P-1 •1) CALL MI23 (PEL1 •PEL2 •32-3 •3) 615 CONT INUE 3P1=PEL2 •1 C IF (STCNT(SP1) •N = 0) GO TO 1155	0009 0005 0009 0009 0009 0009 0009
CALL M123 (PEL1 .PEL2 .32-2 .2) GD TO 615 613 CONTINUE PEL2=1 43 (DTBUF(1.DTCOD) .P-1 .1) PEL1=1AB(.DTBUF(1.DTREF) .P-1 . 3) CALL M123 (PEL1 .PEL2 .32-3 .3) 615 CONTINUE 371=PEL2+1	00009 00009 00009 00009 00009 00009

	00009
C BEGIN DECOJE LOOP: RETRIEVE NEXT CODE WORD LENGTH (L)	00009
1000 CONTINUE	00009
L= 0	00009
LPACK= 0	99999 00009
1002 LENRIT =CJOS(1.INDEX.T)	00009
IF(DIAG) WRITE(6.1003) LENBIT.MODE.LBITS.L	00009
1003 FOSMAT (216. 712. I6)	00009
GD TO (1040,1200,1205,1190), NODE STOP 1040	90009 90000
-1040 CONTINUE	
IF(LBITS.EQ.CODS(3.INDEX.T)) GD TO 1100	00009
C NO WATCH; ADVANCE CODE WORD INDEX VIA DECODE THREAD	00009
INCEX#CO)S (2 a INDEX a T)	00009
IF (INDEX. GZ. 67. AND. T. EQ. 1) GO TC 1190	00010
IF(INDEX.GE.66.AND.T.EQ.2) GO TO 1190	00010
IF(CODS(1.INDEX.T).EQ.LENBIT) GO TO 1040	00010
C CODE WORD LONGER; FROM THE TOP	00010 00010
C CODE NORD EDNOSKY FROM THE 1CP	00010
GO IC 1002	00010
C MATCH FOUND	00010
L 100 CONT INUE	
CDELP=CDELP+L	00010 00010
C NOT AN EOL	00010
C NOT AN EUL	00010
RUN= IN CEX GC TC (1110 -1120 -1130 -1130 -1130) - T	00010
	00010
C 1 · 2 3 4 5 6	00010
STCP 1100	0001 C
ILLIO CONTINUE	00010
IF(INDEX.E3.66) GO TO 1140 IF(INDEX.E3.65) RUN=PELMAX+1	0001C
GO TC 1150	0001¢
IF (INDEX . EQ .65) GO TO 1140	00010
GO TO 1150 1130 CONTINUE	0001¢
IF(INDEX.E).33) GD TO 1140	00016
GO TO 11:50	00010
L=0	00016
LPACK=C	- 3001 € 0001(
L'ENA IT=11	00010
IF (DIAG) WRITE(6,1003) LENBIT, MODE, LBITS, L	0001
GO TO (1145.1200.1205.1190).MODE	00010
STOP 1145 1145 CONTINUE	00010
	00010
RUN=LBITS 1150 CONTINUE	00010
STCNT(SPI)=RUN	00010
1155 CDNT INUE IE(STCNT(SP11=11_1190.1160.1165	00010
C .	00010
C INCORRECT PREDICTION	0001
1160 CONTINUE	00010
IF(NPRED(SPI)) 1161.1163.1162	00010
II62 CALL MI23(1,0TBUF(1,0TCOD),0TELP.1)	00010
1163 SICNI(SPL)=0 GO TO 1170	0001
C	0001
C CORRECT PREDICTION	
1 165 CONTINUE	00010 00010
IF (PREDCT (SPI)) 1161,1168,1167	00010
1167 CALL MI28(1.0TBUF(1.0TCOD).OTELP.1)	0001(

JNC_ASSIFIED

TIBE STONY(SPI)=STONY(SPI)=I	000106
1170 OTELP=OTELP+1	401000
7000 CONTINUE 1F(DIAG) #RITE(6.1185) (CTBUF(1.CTCOD).I=1.60)	000100
1185 FORMAT (6210)	000106
	0001 00
C CHECK FOR ERRORS	000107
C DO 1175 1-1.16	000107
ĬĔ(ŠŤĊŇŤ(ĨĴ.NĔ.O.ANDASTCNT(I).NE.1) GO TO 1190	000107
1175 CONTINUE	000107
STATUS=1	000107
RETURN	-000107
C NO MATCH FOUND IN CODE TABLE (3)	000107
C	000107
TIGO CONTINUE	90100
ST ATUS=3	_000108
SE TUGN.	000108
C ECLI DETECTED (4)	000106
	-00010E
1 20 0 CONT INUE	000106
STATUS =4	000105
RE TURN	000108
E FOLZ DETECTED (5)	_ ocotos
C	000109
1205 CONTINUE	000105
- 374 TU3=5	000105
FETURN E N D	00010
SUBROUTIVE GCODER(LENGTH, POLAR, CDELCY, CODATA)	000105
C	000105
IMPLICIT IN IEGER (4-2)	000105
COMMON/BJFF/PELBJF(60,2),CDBUF(240), OTBUF(60,2),STFBUF(240), STAT(3000)	00011
	-00011€
CTABLE(16).CSTART(16).STBUF(1728).STRUN(1728)	000110
COMMEN ÆR AY / ERRORS (2500)	711000 711000
	OUULL
TO THE TOTAL PROPERTY OF THE TOTAL PROPERTY	
Cossessessessessessessessessessessessesse	00011
Ce************************************	00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH	000110
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE-0	000110 000110 000110
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0	000111 000111 000111 000111
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0	000110 000110 000110
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS	000110 000110 000110 000110 000111 000111
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS C IF (POLAR LT of oDR 20 LAR GT of 2) CALL EXIT	00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCDE=0 MLENG=0 C CHECK INPUTS C IF (POLAR_LT_01_OR_20LAR_GT_2) CALL EXIT IF (LENGTH-LT_0.0.) R. LENGTH-GT_128) CALL EXIT	00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE. MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS C IF (POLAR LT -1 - OR - FOLAR GT - 2) CALL EXIT IF (LENGTH - LT -0 - OR - LENGTH - GT - 1728) CALL EXIT C	00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS C IF (POLAR_LT_01_OR_2DLAR_GT_2) CALL EXIT IF (LENGTH-LT_00_OR_LENGTH-GT_01728) CALL EXIT C IF (LENGTH-LT_00_OR_LENGTH-GT_01728) CALL EXIT	00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS C IF (POLAR_LT_01_OR_2DLAR_GT_2) CALL EXIT IF (LENGTH-LT_00_OR_LENGTH-GT_01728) CALL EXIT C IF (LENGTH-LT_00_OR_LENGTH-GT_01728) CALL EXIT	00011 00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS C IF (POLAR LT -1 - OR - 3 DLAR - GT - 2) CALL EXIT IF (LENGTH - LT - 0 - OR - LENGTH - GT - 1728) CALL EXIT C TP (LENGTH - LE - 603) 60 TO 10 C CALCULATE 4AKE UP CODE INDEX, CODE, LENGTH C AND WRITE TO CODE LINE	00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS C IF (POLAR_LI_1_OR_2OLAR_GI_2) CALL EXII IF (LENGTH-LI_0O_DR_LENGTH-GI_1728) CALL EXII C TP (LENGTH-LI_0O_DR_LENGTH-GI_1728) CALL EXII C C CALCULATE 1AKE UP CODE INDEX, CODE, LENGTH C AND WRITE TO CODE LINE	00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS C IF (POLAR_LI_01_OR_20LAR_GI_2) CALL EXII IF (LENGTH-LI_00_DR.LENGTH-GI_1728) CALL EXII C IP (LENGTH-LI_05) 30 TO 10 C CALCULATE 4AKE UP CODE INDEX, CODE, LENGTH C INCEX=LEGITM/64-64	00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS C IF (POLAR_LI_1_OR_2OLAR_GI_2) CALL EXII IF (LENGTH-LI_0O_DR_LENGTH-GI_1728) CALL EXII C TP (LENGTH-LI_0O_DR_LENGTH-GI_1728) CALL EXII C C CALCULATE 1AKE UP CODE INDEX, CODE, LENGTH C AND WRITE TO CODE LINE	00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS IF (POLAR_LI_1_OR_2OLAR_GI_2) CALL EXIT IF (LENGTH-LI_0O_DR_LENGTH.GI_1728) CALL EXIT C IF (LENGTH-LI_0O_DR_LENGTH.GI_1728) CALL EXIT C C CALCULATE 4AKE UP CODE INDEX, CODE, LENGTH C AND WRITE TO CODE LINE INCEX=LEGITH/6A +6A MCODE=CODE(1, INDEX, POLAR) MLENG=CODE(1, INDEX, POLAR) CALL MIRCH 1281 14005 CODUT COELCT +1 MLENG)	00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS C IF (POLAR_LT_01_OR_2OLAR_GT_2) CALL EXIT IF (LENGTH-LT_00_OR_LENGTH-GT_01728) CALL EXIT C IP (LENGTH-LT_00_OR_LENGTH-GT_01728) CALL EXIT C C CALCULATE 4AKE UP CODE INDEX, CODE, LENGTH C AND WRITE TO CODE LINE C INCEX=LEGITH/64*64 MCODE=CODE(1, INDEX, POLAR) MLENG=CODE(1, INDEX, POLAR) CALL MI28(1+CODE_CODDUT, COELCT_1) MLENG) CODELCT=COELCT+MLENG	00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 MLENG=0 C CHECK INPUTS IF (POLAR_LT=1 **OR**********************************	00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 MLENG=0 C CHECK INPUTS IF (POLAR_LT=1 **OR**********************************	00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS C IF (POLAR_LT_1_OR_2OLAR_GT_2) CALL EXIT IF (LENGTH-LT_0.0) OR.LENGTH.GT -1728) CALL EXIT C TP (LENGTH-LE_0.0) OD TO 10 C CALCULATE 4AKE UP CODE INDEX, CODE, LENGTH C AND WRITE TO CODE LINE C INCEX=LE_4GTH/64 64 MCODE=CO) E(3, INDEX.POLAR) MLENG=CO) E(1, INDEX.POLAR) CALL MIZE (1005_CODUT_COELCT +1, MLENG) CDELCT=CO ELCT+MLENG CDEATA=CODATA+MLENG	00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CCDE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS IF (POLAR_LI_1_OR_2DLAR_GI_2) CALL EXIT IF (LENGTH-LI_0.0-) OR LENGTH-GI_1728) CALL EXIT C IP (LENGTH-LI_0.0-) OR TO TO C CALCULATE 4AKE UP CODE INDEX, CODE, LENGTH C AND WRITE TO CODE LINE INCEX=LI_4GTH/64 *64 MCODE=CODE(3,INDEX,POLAR) MLENG=CODE(1,INDEX,POLAR) CALL MISS(1CODE) CODUT-CODUT-COLOR CODE(CT-LI_NHENG) CDELCT=CODELCT+MLENG C CALCULATE TERMINATING CODE INDEX, CODE, LENGTH C AND ACD TO CODE LINE	00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS IF (POLAR_LT = 1 **OR ********************************	00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS C IF (POLAR_LT.1 = OR_2OLAR_GT.2) CALL EXIT IF (LENGTH *LE***********************************	00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS IF (POLAR LIT = OR = 20 LAR = GT = 2) CALL EXIT. IF (LENGTH = LT = 0.03 + 0.0 TO 1.0 C CALCULATE = 4AKE UP CODE INDEX, CODE, LENGTH AND WRITE TO CODE LINE C INCEX=LEGITH/6A +6A MCODE=CODE(3 = INDEX + POLAR) MLENG=CODE(1 = INDEX + POLAR) CALL MIES = (1005 + CODE) + CODE LENGTH C CALCULATE TERMINATING CODE INDEX + CODE, LENGTH C CALCULATE TERMINATING CODE INDEX + CODE, LENGTH C CALCULATE TERMINATING CODE INDEX + CODE + LENGTH C CALCULATE TERMINATING CODE INDEX + CODE + LENGTH C CALCULATE TERMINATING CODE INDEX + CODE + LENGTH C CALCULATE TERMINATING CODE INDEX + CODE + LENGTH C CODE = CODE(3 + INDEX + POLAR) TLENG=CODE(1 + INDEX + POLAR) TLENG=CODE(1 + INDEX + POLAR) TLENG=CODE(1 + INDEX + POLAR)	00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS IF (POLAR_LI_1_1_OR_2OLAR_GI_2) CALL EXII. IF (LENGTH-LE-03) SO TO 10 C CALCULATE MAKE UP CODE INDEX, CODE, LENGTH C AND WRITE TO CODE LINE C INCEX=LE-GIM/6A 6A MCODE=CODE(3, INDEX, POLAR) MLENG=CODE(1, INDEX, POLAR) CALL MI20 (CODE) CODE INDEX, CODE, LENGTH C CALCULATE TERMINATING CODE INDEX, CODE, LENGTH INDEX=MOD (LENGTH-6+)+1 TCODE=CODE(1, INDEX, POLAR) TLENG=CODE(1, INDEX, POLAR) CALL MI23 (TODE_CODE) CODE(T+1, TLENG)	00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS IF (POLAR_LT.1 **OR.20LAR.GT.2) CALL EXIT IF (LENGTH.LT.0.) JR. LENGTH.GT.1728) CALL EXIT C P(LENGTH.L.2.65) 80 TO 10 C CALCULATE 4AKE UP CODE INDEX, CODE, LENGTH AND WRITE TO CODE LINE C INCEXELEGITI/64.64 MCODE=CODE(1, INDEX.POLAR) MLENG=CODE(1, INDEX.POLAR) CALL MISS (**CODE, CODE) CODE INDEX, CODE, LENGTH C CALCULATE TERMINATING CODE INDEX, CODE, LENGTH AND ACD TO CODE LINE C CALCULATE TERMINATING CODE INDEX, CODE, LENGTH AND ACD TO CODE LINE 10 CONTINUE INDEX.MODILENOTH.GO) TLENG=CODE(1, INDEX.POLAR) TLENG=CODE(1, INDEX.POLAR) CALL MISS (**CODE, CODE) TLENG=CODE(1, INDEX.POLAR) CALL MISS (**CODE, CODE) CODELCT=CODE(1, INDEX.POLAR) CALL MISS (**CODE, CODELCT+1, TLENG) CODELCT=CODE(T+1, TLENG)	00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 MLENG=0 C CHECK INPUTS IF (POLAR_LIT_1_OR_3OLAR_GIT_2)_CALL_EXIT_ IF (LENGTH *LIT_0_) TR.LENGTH *GT *1728) CALL_EXIT_ C THE (LENGTH *LIT_0_) TR.LENGTH *GT *1728) CALL_EXIT_ C CALCULATE 4 AKE UP CODE INDEX, CODE, LENGTH C AND WRITE TO CODE LINE INCEX=LEIGTH/6A *6A ***CODE=CODE(1, INDEX, POLAR) MLENG=CODE(1, INDEX, POLAR) CALL_MISSINGEX_CODUF* *COSELCT***INHENG**) CODELT=CODE_CT***ILENG CODEATA=CODATA***ILENG COLOUTINUE 10 CONT INUE 10 CONT INUE 10 CODE CODE(1, INDEX, POLAR) TLENG=CODE(1, INDEX, POLAR) TLENG=CODE(1, INDEX, POLAR) CALL_MISSINGEX_POLAR) CALL_MISS	00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 C CHECK INPUTS IF (POLAR_LI_1_1_OR_2DLAR_GI_2) CALL EXII. IF (LENGTH LI_0_0_)R_LENGTH.GI_1728) CALL EXII C IF (LENGTH LI_0_0_) GO TO 10 C CALCULATE 4AKE UP CODE INDEX, CODE, LENGTH C AND WRITE TO CODE LINE INCEX_LEGITH/6A 6A MCODE=CD_E(I_1 INDEX, POLAR) MLENG=CD_E(I_1 INDEX, POLAR) CALL MISS(CIT+ALENG CDEATA=CD_DATA+MLENG C CALCULATE TERMINATING CODE INDEX, CODE, LENGTH C AND ACD TO CODE LINE 10 CONT INUE INDEX_MODITATION (CODE INDEX, CODE, LENGTH C CALCULATE TERMINATING CODE INDEX, CODE, LENGTH C CODE=CD_E(I_1 INDEX, POLAR) TLENG=CD_E(I_1 INDEX, POLAR) CALL WISS(TODE_CODE(I_1 INDEX, POLAR) CODE(I_1 TODE_CODE(I_1 INDEX, POLAR) CODE(I_1 TO	00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 MLENG=0 C CHECK INPUTS IF (LENGTH.LT.1.OR.2DLAR.GT.2) CALL EXIT. IF (LENGTH.LT.0.OR.LENGTH.GT.1728) CALL EXIT C IP (LENGTH.LE.65) OD TO 10 C CALCULATE MAKE UP CODE INDEX, CODE, LENGTH C AND WRITE TJ CODE LINE INCEX=LEMTHAGA 6A MCODE=CD) E(3, INDEX.POLAR) MLENG=CD) E(1, INDEX.POLAR) CALL WIES (MCODE-CODELTT.MLENG) CDELCT=CD ELCT+MLENG C CALCULATE TERMINATING CODE INDEX, CODE, LENGTH AND ACD TJ CODE LINE 10 CONT INUE INDEX=CDJE(3, INDEX.POLAR) TLENG=CDJE(1, INDEX.POLAR) TLENG=CDJE(1, INDEX.POLAR) TLENG=CDJE(1, INDEX.POLAR) CALL WIES (TODE-CODELT+I, TLENG) CDELCT=CDECT+ILENG C RETURN	00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 MLENG=0 C CHECK INPUTS C IF(POLAR_LIT_1_OR_2DLAR_GI_2) CALL EXIT. IF(LENGTH+LIT_0.0) 30 TO 10 C CALCULATE 4AKE UP CODE INDEX, CODE, LENGTH C AND WRITE YO CODE LINE C INCEX_LEGIH/6A+6A MCODE=COJE(3,INDEX,POLAR) M.ENG=COJE(1,INDEX,POLAR) CALL MIRB(16005_CODE) CODE INDEX, CODE, LENGTH C CALCULATE TERMINATING CODE INDEX, CODE, LENGTH C CALCULATE TERMINATING CODE INDEX, CODE, LENGTH C CONTINUE 10 CONTINUE 1102E-COJE(1,INDEX,POLAR) TCODE=COJE(1,INDEX,POLAR) TLENG=COJE(1,INDEX,POLAR) CALL MIRB(LENGTH,64)+1 TCODE=COJE(1,INDEX,POLAR) TLENG=COJE(1,INDEX,POLAR) CALL MIRB(CODE) COBUF,COELCT+1,TLENG) CODELCT=CJELCT+TLENG C PETURN SURROUTINE STUFF!(CDBUF,STFBUF,STFBIT,COELCT)	00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH MCODE=0 MLENG=0 C CHECK INPUTS IF (LENGTH LT.1.0R.20LAR.GT.2) CALL EXIT IF (LENGTH LT.0.0.3R.LENGTH.GT.1728) CALL EXIT C IP (LENGTH LE.65) 30 TO 10 C CALCULATE 4AKE UP CODE INDEX, CODE, LENGTH C AND WRITE TO CODE LINE INCEXELE (GTM/AA MAA) MCODE=CO) E(1, INDEX.POLAR) MLENG=CO) E(1, INDEX.POLAR) COLCT=COELCT+MLENG COLCT=COELCT+MLENG C CALCULATE TERMINATING CODE INDEX. CODE, LENGTH AND ACD TO CODE LINE C CALCULATE TERMINATING CODE INDEX. CODE, LENGTH AND ACD TO CODE LINE C CALCULATE TERMINATING CODE INDEX. CODE, LENGTH END ACD TO CODE LINE C CALCULATE TERMINATING CODE INDEX. CODE, LENGTH END CODE CODE(1, INDEX.POLAR) TLENG=CODE(1, INDEX.POLAR) TLENG=CODE(1, INDEX.POLAR) CALL WIST (TODE.COBUF, COELCT+1, TLENG) COELCT=COELCT+TLENG C RETURN SUBROUTINE STUFFI(COBUF.STFBUF, STFBIT.COELCT) INGLICIT (TITEGER(A-Z))	00011 00011
C INITIALIZE MAKE UP CODE, MAKE UP CODE LENGTH C MCODE=0 MLENG=0 MLENG=0 C CHECK INPUTS C IF(POLAR_LIT_1_OR_2DLAR_GI_2) CALL EXIT. IF(LENGTH+LIT_0.0) 30 TO 10 C CALCULATE 4AKE UP CODE INDEX, CODE, LENGTH C AND WRITE YO CODE LINE C INCEX_LEGIH/6A+6A MCODE=COJE(3,INDEX,POLAR) M.ENG=COJE(1,INDEX,POLAR) CALL MIRB(16005_CODE) CODE INDEX, CODE, LENGTH C CALCULATE TERMINATING CODE INDEX, CODE, LENGTH C CALCULATE TERMINATING CODE INDEX, CODE, LENGTH C CONTINUE 10 CONTINUE 1102E-COJE(1,INDEX,POLAR) TCODE=COJE(1,INDEX,POLAR) TLENG=COJE(1,INDEX,POLAR) CALL MIRB(LENGTH,64)+1 TCODE=COJE(1,INDEX,POLAR) TLENG=COJE(1,INDEX,POLAR) CALL MIRB(CODE) COBUF,COELCT+1,TLENG) CODELCT=CJELCT+TLENG C PETURN SURROUTINE STUFF!(CDBUF,STFBUF,STFBIT,COELCT)	00011 00011

C	COMMON /3 3281 T/MASK (32) .COMASK (32) .LIBIT (32) .LZBIT (32)	0001
	INTEGER WASK COMASK LIBIT LIBIT	0001
	IN171AC12E-9776UF-70-6	- 000 1 000 1
		0001
	00 50 [=2,240 SYFBUF(1)=3	0001
50	CONTINUE	0001
	1=32+1+15	0001
	J= I - STFBUF (1) = CCBUF (1)	0001
ŗ		
	PICK UP EL	0001
	LSS=148(C08UF(2)+1+13)	0001
1 00	CALL MIZH (LSS-STFRUF(2)-1-13)	0001
	POL= 148(C08UF. I. 1)	0001
	- IF 1PCL VE3V1) - 38 - 70 - 110	0001
***	GO TO 150	0001
110	LICNT=0	0001
1 50	CALL MIZH(ZZLASTERNEALAL) CONTINUE	_0001
1 30	I= I+1	000 1 000 1
	J=J+1 IF(LICNT.LE.9) GD TD 200	-0001
	CALL MIZE (1.STFBUF.J.i)	0001
	L 1 CN 7 = 0 J = J + 1	0001
		نموم
1	TEST IF FINISHED	0001
500		- 000 I
	IF(I•L∃•CD±LCT) GD TD 100 _STEBIT=J-1-CD±LCT	0001
	CDELCL=1- I	0001
	RE TURN	0001
	BL CCK DAT A	0001
	- 14PL 10 17 - 14 1EGER (A-2)	0001 - 0001
* ** *	**************** FILE DEFINITIONS *************	0001
	COMMON/FILES/TERM.LPFIL.PELFIL.OTFIL.ERFIL	0001
	COMMEN /BUFE /PEL BUF (60 . 2) . CORUF (240) .	0001
	* OTBUF(60.2).STFBUF(240). STAT(3000)	0001
	COMMON AUFF/CODE(3,92,2),CCDS(3,68,6),FREDCT(16),NPRED(16),	0001
	CCMMCN/ER AY/ERRORS (2500)	0001
	************** LABELLED COMMON VARIABLES *************	0001
	COMMONZIVARZPELMAX.VRES.EPHASE.CMPMAX.ERRMOD.LINMAX.K COMMONZEVARZINENNO.CTENNO.DTELW.INELP.CDELP.DTELP.CDELW.	0001
	COELCT . NOLCT . TOGATA . TODEL . ERRPHT . ERRLIM.	0001
	# ERRCNT : INC & T.CONSEC .LNNCBF .ZCNT .#RJBUF ,LPACK,	0001
	COMMON/ICHAR/DD.II.MM.TT.NN.YY	- 0001 0001
	COMMON/LDGIC/SEARCH, DIAG, SYNC, LSS, WRITE, CHCOL, ONE LCGICAL SEARCH, DIAG, SYNC, LSS, WRITE, CHCOL, ONE	0001
		0001
	DATA PRED /1.0.1.0.1.0.1.1.1.1.0.1.0.1.0.1.0.1.0.1	0001
	CATA CTABLE/1.3.4.6.6.6.6.5.6.6.5.6.6.4.3.2/	0001
	- DATA-CSTART/66vtv2v2v2v2vtv2v2vtv2v2v2vtv2/	1000 1000
	CATA TERA.LPFIL.PELFIL.OTFIL.ERFIL/5.6.1.2.3/	0001
	DATA DD.11.44.TT.NN.YY/DT.TT.AV.TT.TV.TY// DATA PELAAX.VRES, EPHAS E. CMPMAX.ERRMOD.LINMAX/1728.2.0.96.TT.3000	0001
	DA1A K /2 /	_0001
:	CATA DIAG/.FALSE./	0001
	- DATA CODE(1+ 1+1) +CODE(2+ 1+1) + CODE(3+ 1+1) + U+ 70+20035/	-0001
	DATA COJE(1: 2:1).CODE(2: 2:1).CODE(3: 2:1)/ 6: 90:20007/ DATA COJE(1: 3:1).CODE(2: 3:1).CODE(3: 3:1)/ 4: 4:20007/	000 l
	DATA CDJE(1, 4,1) .CDDE(2, 4.1) .CDDE(3, 4.1) / 4, 5.2000B/	0001
	DATA COPE(1. 5.1).CODE(2. 5.1).CODE(3. 5.1)/ 4. 6.2000B/	0001

DAYA CODELL.	6.1).CODE(2.	6.11.CDDE(3.	8.777 4.	7.200007	00012:
DATA CODELLA	7.11.CODE(2.	7.1).CODE(3.	7.11/ 4.	8.Z000E/	00012
DATA CODE(1.	9.1).COCE(2.	9.1) .CODE (3.	9.11/ 5.	10.Z0013/	00012.
DATA COJECTA	10.1).CODE(2, - 11.1).CODE(2,	10.1).CDDE(3. -11+1++CCDE(3+		11,20014/	00012.
DATA CODECIA	12.1).CODE(2.	12.1).CODE(3. 13.1).CCDE(3.	12.11/ 5.		00012.
DATA CODE !!	14,11,CODET 2.	14.17 .CODE (3.	14.117 6.	15.200037	00012
DATA CODE(1.		15.1), CODE(3.		16.20034/	00012
DATA COJECL.	17.1), CODE(2.	17.11.CODE(3.	17.11/ 6.	18.Z002A/	00012.
		18.1).CODE(J. -19.1).CODE(J.			00012
DATA COJECT.	20 -1) - CODE (2 -	20\1).CODE(3. 21.1).CODE(3.	20.11/ 7.	21.Z000C/ 22.Z0008/	00012
DATA CODE(1.	22.11.CUDE(2.	22.11.CODE(3.	22.11/ 7.	23.20017/	00012
DATA COJE(1.	23.1).CODE(2.	23.11.CCDE(3. 24.1).CCDE(3.	23.1)/ 7.	24,20003/ 25,20004/	00012
DATA CODELL.	25.1).CJDE (2.	25.1).CCDE(3.	25.11/ 7.	26, Z0028/	00012
DATA COJELL.	-27-11-CUDE(2-	26,1),CCDE(J, -27,1),CDE(J,	27v11/ /·	20,2002B/	
DATA CODE(1.	28.1).CDDE(2.	28.1).CODE(3. 29.1).CODE(3.	20.13/ 7.	29.20024/ 66.20018/	00012
DATA CODE(1)	30.17.CDE(2.	30.17.CODE (3.	30.117 8.	31.200027	00012
DATA CODELLA		31,1),CODE(3, 32,1),CODE(3,		32.Z0003/	00012:
DATA CODE(1.	33,1).CODE(2.	33,1),CODE(3,	33.11/ 8.	34 , 2001 B/	00012
	-35 v11 vc 30E12v	34,1),CDDE(3. - 35,1),CDDE(3.	35 v1 1/ 8 v	-36 v20013/	00012:
DATA CODE(1.	36.1).CODE(2.	36.1).CODE(3. 37.1).CCDE(3.	36.1) / 8.	37.20014/ 38.20015/	00012 00012
DATA CODELLA	38.17.CUDE (2.	39.11 .CODE (3.	38.177 5.	39.200167	00012
DATA CODE(1.		39.1).CDDE(J.		40 . Z 0017/ 41 . Z0028/	00012
DATA CODECL.	41.1), CODE(2.	41.1).CODE(3.	41.13/ 8/	42,Z0029/	00012
DATA CODELLA		42.11.C00E(3. 43.11.C00E(3.		43.2002A/ *** 2002B/	00012
DATA CODELL.	44.11.CODE (2.	44.1), CDDE(3, 45.1), CODE (3.	44.1)/ 8.	45.2002C/ 46.2002D/	00012
DATA CODE(I.	46.11.CDDE(2.	46.1).CDDE(3.	46.1) / 8.	47.20004/	00012
DATA COUE(1.	47.1) .CJDE(2.	47.1).CCDE(3.	47.11/ 0.	43.20005/ 49.2000A/	00012
DATA CODE(1.	49 .1) . CODE (2 .	49.1).CCDE(3.	49.11/ 8.	50. Z000B/	00012
*1)ECCO ATAG		50.1),CCDE(3. 51.1),CCDE(3.		51 •20052/ -52•20053/	00012
DATA CODE(1.	52.1) .CODE(2.	52.1).CODE(3. 53.1).CODE(3.	52.11/ 8.	53,20054/ 54-20055/	00012
DATA CODE(1.	54.1),CJDE(2:	54.[],CCDE(3.	54.1)/ 8.	55.Z0024/	00012
DATA CODE(1.	56-11-CDDE12-	55.1).CODE(3. 56.1).CDDE(3.	50011/ He	57.Z00584	00012
DATA CODE(1.	57.1).CODE(2.	57.1).CODE(3.	57.11/ 8.	58.20059/	00012
ATA CODE(I)	-59+1++C30E12+	58.1).CODE(J. 59.1).CCDE(J.	-53-11/-01	59,2005A/ 59,2005B/	00012
DATA CODE(1.	60.1).CDDE(2.	60.1).CODE(3. 61.1).CODE(3.	60.11/ 8.	61.Z004A/	00012
DATA CODELL.	62. I) . CODE (2.	62.11.CODE (3.	62 . 1) / 8 .	63.20032/	21000
DATA CODE(1,	63,11,CJDE(2,	63.1).CDDE(3. 64.1).CDDE(3.	03.11/ 8.	54,20033/ 5x,20034/	21000
DATA COJE(1.	65.1).CDDE(2.	65.1).CODE(3.	05,1)/ 5,	60.2001B/	21000
	- 67+1)+638£12 +	67v+}v€006(-3 v	-67+1) / 6 +	~~ 2 v2 001 7/ ~~	
DATA CODE(1.	68.1).CODE(2.	68.1),CODE(3.	59.11/ 8.	1.2003//	00012
DATA CODETT.	70.11.CJDE(2.	70.1), CODE (3.	70.1) / 3.	71.20037/	00015
DATA CODECIA	72.1).CODE(2.	71.1),CCDE(3. 72.11.CDDE(3.	72.11/ 3.	73.70065/	00015
DATA CODE(1.	. 73 at 1 aCDDE (2 a	73.1).CODE(3. 74.1).CODE(3.	73.1)/ 8.	74.20068/	00012
-0474-603-11-	75 v 1 -) v C 30C (2 v	- 75++++0000+3+	~75+1}/~~	-70+20000/	00012
DATA COUE(1.	77.11.CODE(2	76.1).CDDE(3. 77.1).CODE(3.	77.11/ 9.	73.Z00D2/	00012
DAYA CODETT	73.13.CJDE (2	78.1).CCDE(3. 79.1).CODE(3.	76.17/ 9.	79.Z00D3/	00013
DATA CODELL.	80.11.CJDE (2)	BO.L.L.CODELL.	-80-TTC-80	B1.Z0005/	00017
DATA CODELLA	81.1).CODE(2:	81.11.CODE(3. 82.1).CODE(3.	31.11/ 9.	82.Z0006/	00013
**************************************	r -93v11- vC 0DE12 -	63-11-11-606-63 -	ー・ジョー・ナー・シャ	~~>++20000 /~~~	
DATA CODELL.	. 85.1).CDCE(2	84.1).CODE(3. 85.1).CODE(3.	85.1)/ 9.	80.Z00DA/	00013
DATA CIDECI.	86.1).CDDE(2	86.11.CODE(3. 87.1).CODE(3.	36.11/ 9.	97.Z003B/	00013
DATA CONCIL	, _, , , , , , , , , , , , , , , , , ,				

DATA CIDECT		. 83.11.CCDE(3		~~~	
DATA CODELL.					00013
DATA CODELL	90.11.CIDE (2	90.1).CDCE(3	a Vialle Ga	13.70018/	00013
DATA CODE(1.				92.Z009B/	00013
DATA CODE(1.				93.20002/	00013
DATA CODE(1.				-65 / 20037/ 6 • 20002/	00013
DATA CODE(1.				4. Z 0 0 0 3/	00013
DATA CODELL.				5.200027	00013
DATA CODE(1.				2.20003/	00013
DATA COSELLA					
DATA CODE(1.				8.Z0002/ 9.Z0003/	, 00013. 00013
-DATA- CODE(IV		• 9•21•600E(3		+0+20005/	
DATA CODE (1.	10.2).CODE(2	. 10.2), CODE (3	. 10.21/ 6.	11.20004/	00013
DATA CODEL 1.		. 11.2).CODE(3		12.Z0004/	00013.
DATA CODE(1.				13.20005/	00013
DATA CODECIA		. 13.2).CCDE(3 . 14.2).CDDE(3		14.Z0007/ 15.Z0004/	00013
DATA CODE(1.		. 15.2).CCDE(3		16.20007/	00013
DATA COSE(1.	16,2),CODE(2	. 16.2),CODE(3	. 16.21/ 9.	17.Z0018/	00013
		▼ 17 2) ▼ CD DE1 3			
DATA CODE(1.		. 18.2).CCDE(3 . 19.2).CODE(3		19.20018/ 1.20008/	00013 00013
DATA CODECT		. 20.2) . CCDE (3	. 20.21/11.	21.200677	00013
DATA CODE(1.	21,2),CODE(2	. 21.2).CDDE(3	 21,2)/11. 	22 .Z 0068/	00013
DATA CODELL					- 00013
DATA CODE(1.		. 23.2).CODE(3 . 24.2).CODE(3		24.Z0037/ 25.Z0028/	00013 00013
DATA CODE		+ 25+21+CCDE(3		-24-70017/	- 00013
DATA CODE(1.				27.20018/	00013
DATA CODE(1.				28.Z00CA/	00013
DATA CODE(1.			· 28.21/12.	29.200CB/ 30,200CC/	00013
DATA CODECIA		30.21.CDDE(3		31. ZOOCD/	00013
DATA CODE(1.	31.2).CDDE(2	. 31.2).CDDE(3	. 31.21/12.	32.Z0068/	00013
DATA CODE(1.	32.2).CODE(2	. 32.2).CCDE(3	. 32,21/12.	33.Z0069/	00013
**************************************		* 33*2)*CODE(3 . 34.2),CODE(3		-34 √2006A/ 35•2006B/	- 00013 00013
DATA CODE(1.				35.200D2/	00013
DATA CODETI	36,2),0002(2	. 36.2), CODE (3	. 30,2)/12.	37.Z00037	00013
DATA CODE(1.				33.Z00D4/	00013
DATA COSELLA DATA COSELLA		. 38.2).CODE(3		39.Z00D5/ 40.Z00D6/	<u> </u>
DATA CODE(1.				41 . Z00D7/	00013
	- +1 +2) + C 30E (2	* *1+2)+ CD DC(3	v-+1+21/12+	-42 vZ 006C/	
DATA CODE(1.		• 42.2).CODE(3		43,20060/	00013
DATA CODE(I,		. 43.2),CODE(3	· +3,2)/12, · 44,2)/12.	44.Z00DA/ 45.Z00DB/	00013 00013
DATA CODE(I	45.2).CODE(2	45.2).CODE(3	. 45,2)/12,	45 • Z 0 0 5 4 /	00013
DATA CODELL	<u> 46.21.CODE(2</u>	- 46.21. CODE(3	-46.212 12.		نـــــــــــــــــــــــــــــــــــــ
DATA CODE(1.		. 47.21 .CODE (3		46.Z0036/	00013
CATA CODE(1		• 48•2)•CDDE(3 • 49v2)•CCDE(3	• 48.2)/12.	49•20057/ - 30•20064/	£1000
DATA CODE(1.	50.2).CODE(2	50.2).CODE(3	. 50.2)/12.	51 .Z 0065/	00013
DATA CODELLA	51 .2) . CJCE (2	. 51.2).CCDE(3	. 51.21/12.	52. Z0052/	00013
DATA COSELLA		. 52.2) .CCDE (3	• 52.21712.	53.200537	00013
DATA CODE(1,	54.21.0306(2	. 53.2).CODE(3	. 53,21/12,	54.Z0024/	00013
DATA CODE(1.	55.2) CJDE (2	, 55,2),CODE(3	. 35.21/12.	56.20038/	00013
DATA CODE(1:	, 56,2),CJDE(2	. 56 .2) .CODE(3	. 56.21/12.	57.Z0027/	00013
DATA CODE(1)	- 57+21+690E(2	* 57*2) *CODE(3	* 27 v21/12v	-50 v20028/-	
DATA CODELL	59.21.CODE(2	. 59.2) .CODE(3	• 50•21/12•	59120058/ 50.70059/	00013 00013
DATA CODECT	60.21.030212	60.21.CODE(3	60.21/12.	61.200287	00013
DATA CODE(1	• 61•2) •C DOE (2	. 61.2).ccp=(3	. 51.21/12.	52.7002C/	00013
DATA CODELL	62.21.CDDEL2	-62.21.CODE(3	• <u>62•21/12</u> •	63.Z005A/	00013
DATA CODECT.	64.21.CODE(2	. 63.2),CCDE(3 . 64.2),CCDE(3	· 03,21/12.	66.20067/	00013 00013
~9474~633±(1 7	r 65v2)v699E12	v 65v21vCB9E43	v 05v≥}/10v	- 20 y 2 000F /	
DATA CODE(1.	55.2).CJDE(2	. 66.2).CQDE(3	. 00.21/12.	57.200C8/	00013
DATA COUNTIL	67,21,0006(2	. 67,2),CODE(3	· 6/·2)/12·	63.700E9/	00015
DATA CODE(1.	59.2).CODE(2	: 69.2).CODE(3	• 69•21712•	70 • Z 0033/	00013
DATA CODELL	_70 a2) a CODE (2	- 70-21-CODE(3	a 70a2)/12a	71.20034/	00013
DATA CDDE(1.	71.2).CJDE(2	• 71 •2) •C DDE (3	. 71.21/12.	72.Z0035/	00017
-DATA-C33614-	- 7292796JUR(2	72,2),CDDE(3	· (2,2)/13,	/3,2006/	00013
DATA CODE(1.	74.2).CDDE(2	. 74.2).CDDE(3	. 74.21/13.	75.7004A/	00017
DATA CODECT	75.2).CJOE (2	. 75.2).CCDE(3	. 75.21/13.	70. Z 0048/	00013
DATA CODE(L.	75.21.030E(2	?• 76•2)•CCDE(3 • 77•2)•CODE(3	• 70,2)/13,	77 • Z 0 0 4 C /	00013
2010 CO25111					00013

DATA COLE	70.5	1 ~ Z DD 2 Z 3 ~ .	78 21 CCDE(3	~ 7 0.71712	73 788797	00013
DATA CODE			79.2).CODE(3		BU . Z 0073/	00013
	La 80.a2	1.CODE 12.	80 .2) . CODE (J	. 40.21/13.	81.Z00.74/	00017
DATA CODE().CODE(2.	81.2).CODE(3 82.2).CODE(3			00013
			- 83 ₁2 1 1 CC 0E (3		8J.20076/ -34.20077/	00013
DATA CODE	1. 84.2) . CODE(2.	84.2).CDDE(3	. 34.21/13.	85.Z0052/	00013
DATA CODE) .CDDE(2.			86.Z0053/	00013
DATA CODE(1.CODE(2.			37.20054/ 88.20055/	00014
DATA CODE	1. 33.2	1.CODE (2.	88.2) .CCOE (3	· ##*51/17*	32 . Z005A	00014
DATA CODE			89.2).CDDE(3		90.Z005B/	00014
DATA CODE(90,2),CCDE(3 -91,2),COOE(3		→1.20064/ ->2.20065/	00014
DATA CODE(1, 92,2).CJDE(2.	92.2).CDCE(3		93.20002/	00014
DATA COOS) . C J D S (2 .			12.2001 B/	00014
DATA COUS) .CDDS(2.			3.200047	00014
DATA COUST		i coasta.			5.Z000B/	00014
DATA COOST) .CJC\$ (2.			6.Z001F/	00014
DATA CODS(.) •CODS(2• } •CODS(2•			7 • 2001 C/	00014
DATA COOS) .CODS(2.			3.2000A	00014
DATA CODS!	1. 9.1).CJDS(2.	9.11.0005(3	. 9.11/ 5.	10.2000F/	00014
DATA CODSI).CDDS(2.	10717.CCD5 (3		1.20003/	00014
			12.11.00363		13.20018/	00014
DATA COOSI	1. 13.1	1.000512.	13.1),0005(3	. 13.11/ 0.	14.Z0005/	00014
DATA CODS ().CJDS(2.	14.1).CODS(3		15.20001/	00014
DATA CODS (-15,11,0005(3 16,1),0005(3			00014
DATA CODS	1 . 17.1		17.1).0005(3		18.20079/	00014
JATA CJOST			13.17.00513			00014
DATA CODS() • CODS (2 •	19,1),CODS(3 20,11,CDS(3		20.20033/ 21.20025/	00014
DATA COUST). CODS(2.			22.20027/	00014
DATA CODS			22.11.CCD5 (3			00014
			- 23,11,0005(3 24,1),0005(3		24 v20022/ 25 v20024/	 00014
DATA COOS			25,1),0005(3			00014
DAYA COUST	1. 26.1).0005(2,	26.17.00503	. 26.1)/ 7.	27.20023/	00014
DATA CODS(27.1).0008(3		23.Z003A/	00014
			28 <u>.11.</u> CODS(3		29.ZQQ39/	00014
DATA CIDS(1, 30,1	1) . C O D S (2.	30,1),0005(3	. 30.11/ 7.	JI .Z0014/	00014
			-31+11+0009(3			00014
DATA CODS(10 3201 1. 33.1	.).CODS(2.	32,1),CODS(3 33,1),CODS(3	1. 32.1)/ 7. 1. 33.1)/ 7.	33.20015/ 34.20019/	00014
DATA CODST	1. 34.1) .CODSIZ.	34.11.000513	34.177 3.	33.200407	00014
DATA CODS		1) . C JD S (2.				00014
DATA COOSE			36.11.CGD5(3 37.1).CGD5(3		37.200F6/	00014
CATA COUS ().CJOS(2.			39.200F5/	0001+
- 9474 C33 St			-3911 11 CC0515			00014
DATA CODS (1).CODS(2.	40.1).0005(3	. 40.1)/ 6.		00014
שַׁבְּבָבָ אַנְאַלָּ בַּיִ	1, 42,1	17.6333(2.	42.17.000513	32.117 3.		- 700014
DATA CODS(1. 43.1) • C JDS (2•	43.1),0025(3	43.11/ 8.	44.Z0036/	00014
			_44.ll.ccos(3 45.l).coos(3			00014
DATA CODS(1 • 45 • 1) .C JDS(2.	46 ,1) , CCD5 (3	. 46,11/ 8.	47.Z0001/	00014
- - 0 4T A - C3 3 S ({ 	+++0005+2+	-+7+1++600-51-3	V 47+1) / Uv	-+3720031/	- 00014
			48,1),COCS(3 49,1),CCDS(3			00014
CATA CODST	1 . 50 .1	1.0005(2.	50.1).0005(3	. 50.1) / 8.	51.200027	00013
DATA CODS(1. 51.1	1 .CJDS(2,	51.1).CCDS(3	. 51.11/ 9.	52 • Z 00 85/	0001-
DATA COUST	10 520 1	1.0000512.	52.11.C005(3	10 54011/ Wa	54.20159/	00014
			54.11.0005(3			00014
-DATA-CODSI	1 v 55 v 1	1 VC 305 1 2V	-55v1)vc09913	1-55+11+ 4+	50 V 2 V 1 E U/	
			56.1).CCD\$(3 57.1).CCD\$(3			00014
DATA CODST	1 . 53 . 1	7.0305(2.	58 . 1) . CC35 (3	. 58.177 9.	37.200 EC7	50013
DATA CODS(1 59 1	1) . 0 30 5 (2 .	59.1).0005(3	1. 39.11/ 9.	60 ·Z0087/	00014
			_60.11.00513 61.10.00513			QQQ1
			62.1).0005(3			00014
- DATA-COS SI	1 v - 63 v i	+ vc 30312v	63+1-1+6005 (3	**************************************		
			64,1),0005(3 65,1),0005(3			00014
			66.11,00051			00012
			67.1).CODS(3			0001-

DATA COUSTLE	55.11.CUDS(Z.	BB-17-CUDS (3-	58.17/13.	57.200037	
DATA CODS(1.	1.2).005(2.	1.2).CODS(3.	1.2)/ 4.	5.Z000B/	00014
DATA CODS(1.	3.2).CODS(2.	2.21.CDCS(3. 3.2).CDDS(3.	3,21/ 3,	4.Z0004/	00014
DATA CODS(1.	4.2).CODS(2.	4.2).CODS(3.	4.2)/ 3.	1.20006/	00014
DATA - 600 St LV					
DATA CODS(1, DATA CODS(1.	6,2),CDDS(2, 7,2),CDDS(2,	6.2).CDD\$(3. 7.2).CDD\$(3.	6.2)/ 4.	7.20003/ 8.2000F/	00014: 00014
DATA CJJSCI.	8.21.0005(2.	3.27.005(3.	8.217 5.	9.200047	00014
DATA CODS(1.	9.2).CODS (2.	9.2).CODS(3.	9.2)/ 5.	10.Z000A/	00014
	11.2).CDDS(2.		11,2)/ 5.	11.70008/ 65,2001D/	00014
DATA CODS(1.	12.2) .CODS(2.	12.2).CODS (3.	12.21/ 6.	13.Z0002/	00014
DATA COSSIL	14.2) .CODS (2.	-13v2) vC009(3v	13v21 / 6v	15.70010/	
DATA CODS(1.		15,2),CODS(3,		16.20038/	00014
DAYA CODS (1.	16.2) . CUDS (2.	16.21.003513.	16.21/ 7.	17.200017	00014
DATA CODS (1.	17.2).CODS(2.			13.20014/	00014
	19.2) .COCS (2.			20.20007/	00014
DATA COOS(1.		20.2) .CODS(3.			00014
DATA CODS(I	22-21-C005(2-	22,2),0005(3,	22.21/ 7.	23.70056/	00014
DATA CODS(1.	23,2),0005(2,	23.2).CDDS(3.	23,2)/ 7,	24.Z0025/	00014
DATA CODS(1.		24.21.0005(3.		25.200077	00014
DATA COUST		25.2).CODS(3. 26.2).CDDS(3.			00014
DATA CODS(1.	27.2),5305(2.	27.2) .CODS(3.	27.21/ 8.	28 . Z002 B/	00015
DATA CODS(1.	23,2),CJDS(2, -29v2),CJDS(2 v	28,2),CODS(3,	28.2) / 8.	29.Z002F/	00015
	30.2),CDDS(2.				00015
DATA CODS(1.	31 .2) . CDDS (2 .	31,2),CODS(3,	31.21/ 8.	32.Z004E/	00015
DATA CODS (1.	32,2),CDDS(2, 33,2),CDDS(2,	32.2).COD5(3.		33 . 200007	00015
DATA COUSCIA		34.21.0005(3.			00015
	35.2),CODS(2,				00015
DATA CODS(1.	-37 +21+63DS(2+	36.2).CCOS(3.		37.Z00AE/ 39.Z00E6/	00015
DATA CODS(1.	33.2).CJOS(2.	38,2),0005(3,	38,21/ 9,	39.20091/	00015
DATA CODS(1)	39,2),CODS(2,	39.2).CODS(3.	39,2)/ 9,	40 . Z005 D/ 41 . Z 000C/	00015
DATA CODS(1.		41,2),CCDS(3,			00015
	42.21.CDDS(2.	42.21.CODS(3.	42.41/5.	43.Z01CF/	00015
DATA CODS(1.	43.2).CODS(2.	43.2).CCDS(3.44.2).CODS(3.	43,21/ 9.		00015
THE COSTA	-45+2}+C3D3+2+	-+5+2}+CDDS(3 +	45,21/ 9,	46VZ 0152/	0001 €
DATA CODS(1.	46.2).CDDS(2.	46.2).CODS(3.			00015
DATA CDDS(1,				48.2000D/ 49.2015E/	00015
DATA COOS(1.	49,2),CDDS(2,	49,2).CDDS(3,	49.21/ 9.	50 •Z 0055/	00015
DATA CODS(1.	50.2).CDS(2. 51.2).CJDS(2.	50.21.COCS(3.		51.Z0151/ 52.Z0001/	00015
DATA CODS(1.	52.2).CODS(2.	52, 2), CODS(3,	52.2)/10.	53,200A8/	00015
	-53v2) vC00912v				
DATA CODS(1.	54.2).CDDS(2. 55.2).CDDS(2.			56.Z0134/	00015 00015
DATA COOSET.	56,2),0005(2,	56.2) .0005(3.	56.21/10.	57.200017	00015
DATA CODS(1+	57.2).CJOS(2.	57.21.CODS(3. 59.2).CODS(3.	57.21/10. 58.21/10.		00015
	59.2).CJDS(2.				0001:
	60.2) .CJDS(2.				0001:
	61,2),6305(2, 62,2),6305(2,			63,Z0120/	0001 4
DATA CDD3(1.	63,2),0008(2,	63.2),000\$(3.	63,2)/11.	5+ . Z0001/	0001:
DATA CODS(I.	64.2),CDS(2, 65.2),CDS(2,	64.2).CDES(3. 65,2).CCDS(3.		12,20014/	0001:
DATA CODS LL				2.20001/	0001
DATA CODS(1.	2.3),0005(2.	2.3).CCDS(3.	2.31/ 3.	3.20002/	0001
DATA CODS(1,	3,3),CDDS(2, - 4,3),CDDS(2,	3,3),CGD\$(3, -4,3),CGD\$(3,	3,3)/ 3, - 4,3)/ 4,	4 • Z0001/ 	0001:
DATA CODS(1.	5.3) .CODS(2,	5,3),CCD\$(3,	5.31/ 4.	6.Z0001/	0001:
OF A CODS(1.	5,3),CJDS(2,	6,3),CGDS(3,	0.31/ 5.	7.2000F/ 3.2001C/	0001
DATA CODS(1.	8,3),0005(2,	9.3),0005(3.		5.2001C/ 5.20001/	0001
DATA_CCDS(1.		9.3).CCCS(3.	9.31/6.		0001.
	10,3),000\$(2,	11,3),0005(3,			0001:
-DATA-C335(-1-v	-1-2+3+ vC-303+2+	-12+7)+ceos(3+	121317 01	13 v Z 0 0 3 /	0001
	13,3),C005(2, 14,3),C005(2,				0001: 0001
DAYA CODS(1.	13,37,0005(2,	15,37,0005(3,	15,31/ 5.	16.200027	0001
DATA COUS (1.	16.3),0008(2)	16,3),0008(3,	16.3)/ 9.	17.20001/	0001
					

JNCL ASSIFIED

MAYA PRINCE	TYTE PRINCE	17.31.000313.	TY-TTY-0:-	TT. 7881 17	00015
DATA CODS(II	19.3).CODS(2.	18.3).CODS(3.			00015
	18-31-030215-				00015
DATA CODS(1.	20.3).CODS(2. 21.3).CODS(2.	20.3).CODS(3. 21.3).CDDS(3.	21.3)/10.	22.20006/	00015: 00015:
		-22 v 3 I v COOS (3 v			
DATA CODS (1.	23.3).CODS(2.	23.3).CODS(3.			00015
DATA COSSIL	24.3).CODS(2. 25.3).CODS(2.	24.3).C00\$(3. 25.3).C00\$(3.	24.3)/10.	25,20009/	00015
DATA CODS(1.	26.31.CODS (2.		26.3)/10.	27,20025/	00015
DATA COUST IN	27-31 -CODS(2-				00015
DATA CODS (1.	29.3).CODS(2. 29.3).CODS(2.	28.3).CODS(3. 29.3).CCDS(3.	28.31/11.	25.Z0001/ 30.Z03A4/	00015
THE PERSON ATAG	-30v 3) vC003(2v	-30+3) +COOS(3+	J0+317114	-10 1005v 15	
DATA CODS(1.	31.3).CJDS(2. 32.3).CDDS(2.	31.3).CDDS(3. 32.3).CDDS(3.	31.31/11.	32.20011/ 34.20001/	00015 00015
DATA COUSTI		33,3),0005(3,	33.31 / 7.	11.200387	00015
DATA CODS(1.	1.4) .CODS(2.	1.4),CCDS(3.	1.41/ 3.	3.20002/	00015
DATA COUSCIA	3.4).CODS(2.	3.4).CODS(3.	3.41/ 3.	4.20001/	00015
DATA COS(1.	4.4) .CODS(2.	4,4) .CODS(3.	4.41/ 3.	5.Z0007/	02015
9474-693511T				-6-20007/	
DATA CODS(1.	6.4).CODS(2. 7.4).CODS(2.	6.4).CODS(3. 7.4).CODS(3.	6.4)/ 4.	7.Z000C/ 8.Z000D/	00015 00015
DATA CIDS(I.	8.4) .CDDS(2.	8.41.CCDS (3.	8.11/ 5.	3.200017	00015
DATA COSCIE	9.4).CODS(2.	9.4).CODS(3.		10.Z001A/	00015 00015
DATA COSSILA	11.4).CODS(2.	11.4) .CODS(3.			00015
DATA CODS (1.	12.4).CJDS(2.	12.4), CD DS (3.	12.41/ 6.	13.20006/	00015
	-13v4) v6005(2v 14,4).CDDS(2v			-33+20007/ 15.20030/	
DATA CODS(1.	15.4),CJDS(2,		15.4)/ 7.	16.20001/	00015
DATA CODS(1.	16,47.0005(2.	16,4),0005(3,	16.417 7.	17.200087	00015
DATA CODS(1.	17.4).CODS(2. 18.4).CODS(2.	17.4), CODS(3, 18.4), CODS(3,	17.4)/ 7.	18.Z006E/	00015
DATA COUSTI	19.4).CODS(2.	19.4),CODS(J.	19.41/ 8.	20.20062/	00015
DATA CODS(1.	20.4) .CODS(2,	20.4).CCD5(3.	20.41/ 8.		00015
DATA COSSILV		-21v4)v6005(3v 22v4)v6065(3v			00015 00015
DATA CD03(1)	23.4).0005(2.			24,20001/	00015
DATA CODSIL.				25.Z000E7	00015
DATA CODS(1.	25.4) .CUDS(2. -26.4).CUDS(2.			26 • Z00 DF/ 27 • Z002 7/	00015
DATA CODS(1.	27.4) .CJDS (2.	27.41.CODS (3.	27.43/ 9.	28.Z0026/	00015
DATA CODS(1,	28.4).0005(2.			29 . Z002 A/	00016
DATA CODS(1+	2914)16905(21 30.4).C305(2.			31.20028/	00010
DATA CODS(1.	31.4).CODS(2.	31.4).CODS(3.	31.4)/10.	32.20001/	00016
DATA CODS(1.	32.4).CJDS(2. 33.4).CJDS(2.		32.31/11.	14.20036/	00019
TATA COSSIL.		1.51.CCCS(3.		2.20001/	00010
DATA CODEC L.	2.5) . COD \$(2.		2.5)/ 2.	3.20001/	00016
DATA CODS(1+	3.5).CODS(2. 4.5).CODS(2.	3,5),C00S(3, 	3,5)/ 4, ***********	4.20001/ 	00016
DATA CODS(1.	5.5).CODS(2.	5.5).CGDS(3.	5.5)/ 5.	6.Z0004/	00010
PATA COUSTI			6.5)/ 6. 7.5)/ 6.	7.Z0001/ 8.Z0002/	00016
DATA CODSCI.	7.3).CUDS(2.		8.51/ 6.	9.20008/	00016
DATA COUSCIA	9.51.0005(2.	9.51.cons(3.		11.200017	00016
		10.5).COOS(J.		12.20007/	0001c
}4F4-C3b5(-t v	- 12+51+6005(2 +	-12+5 +000 \$13+	-12+5) / · · · · ·	-1-3-2-0001/	
		13.5).COCS(J.			00014
DATA COUSTI	15.31.000512	14.5).CCDS(3. 15.5).CCDS(3.	15.5)/ 5.	16.200197	00016
DATA CDJ 5(L.	16.5) .C JDS(2	16.5).CCDS(3.	15.5)/ 9.	17.20051/	00016
DATA COUSTLA	_17.51.CODS(.2.			_1&+ZDQ01/	
		19.5) .CODS(3.			00016
94f4~6039f fy	- 20 +5) + C393 (2 +	2015}10005 (31	-201517101	~21 v20041/	0001+
DATA CODS(1.	21+51+CUDS(2+ - 22+51+CDDS(2+	21.5),CDDS(3. 22.5),CDDS(3.	- 21,5)/10, -21,5)/11.	22+200A0/ 23+20007/	0001¢
DATA CODSTI	23.51.000512	23.5),0005(3.	53.31711.	33.200061	0001
DATA CODS(1.	- 24,5),CDDS(<i>2</i> ,	24.5) .COJS(J.	24.31/12.	25.20003/	00014
		25.51.CDDS(3.			00016
DATA CODS(L)	27.5),CJ9S(2.	27.51.0005(3.	27.51/12.	78.500091	00010
		28,5)+0008(3 1 29,5),0008(3,			
DATA CODSCL	30.51 .CJDS (2)	30.5).CCDS(3.	30.5)/13.	JL . Z 0012/	00011
DATA CODSCI.	31.5).0005(2.	31,51,6055(3	31.5)/14.	35 . 20001/	0001c
DATA CODS(1)		J2.5).CDDS(3.		34,20001/	00016

UNIA CDJS([, 33,5],CUDS(2, 33,5),CUDS(3, 33,5)/11, 24,20005/	00016.
DATA CDJS(1, 1,6),CDDS(2, 1,6),CDDS(3, 1,6)/3, 3,Z0002/	00016.
DATA CODS(1. 2.6).CDS(2. 2.6).CDS(3. 2.6)/ 1. 1.70001/	90016
DATA CDDS(1, 3,6).CDDS(2, 3,6).CDDS(3, 3,6)/ 3, 4,20001/	00016
DATA CODS(1, 4,5),CDDS(2, 4,6),CDDS(3, 4,6)/ 4, 5,Z0001/	00016
	- 00016
DATA COS(1, 6,6).CODS(2, 6,6).CODS(3, 6,6)/6, 7.Z0019/	00016
DATA CODS(1, 7,5),CDDS(2, 7,6),CDDS(3, 7,6)/ 6, 8,Z0001/	00016
DATA COS(1, 8,6),CODS(2, 8,6),CCDS(3, 8,6)/6, 9,20002/	00016
DATA CODS(1. 9.6).CDDS(2. 9.6),CDDS(3, 9.6)/ 7, 10.Z0031/	00016
DATA COUNCIL 10.6) CODS(2, 10.6) CODS(3, 10.5)/ 7, 11.70034/	00016
DATA CODS(1, 11.6).CDDS(2, 11.6).CDDS(3, 11.6)/ 7, 12.Z0030/	00016
DATA CDS(1, 12.6), CDDS(2, 12.6), CCDS(3, 12.6)/ 7, 13, 20039/	00016
DATA C333(17 13/6) YCOD3(27 13/6) YCOD3(37 13/6) / 77 14 YZOD3(/-	
DATA CODS(1, 14.6), CODS(2, 14.6), CDDS(3, 14.6)/ 7, 15,20038/	00016
DATA CDS(1, 15.6), CODS(2, 15.6), CODS(3, 15.6)/ 7, 16.20037/	00016:
DATA COUSTI 16.6), CODS(2, 16.6), CODS(3, 16.6)/ 7, 17.2003A/	00016
DATA CODS(1, 17.6) CODS(2, 17.6) CCCS(3, 17.5)/ 7, 33.20006/	00016
DATA COSCIL 18.6). CODS(2. 18.6). CODS(3. 18.6) / 8. 19. Z0001/	<u>0001</u> &:
DATA CODS(1, 19.6).COCS(2, 19.6).CODS(3, 19.6)/ 8, 20.Z000F/	00016
DATA CDS(1. 20.6).CDS(2. 20.6).CDS(3. 20.6)/ 8. 21.20076/	00016
DATA CODS(1, 22.6).CDDS(2, 22.6).CDDS(3, 22.6)/ 9, 23.Z00D9/	00016
DATA CDS(1, 23,6),CDDS(2, 23,6),CDDS(3, 23,6)/ 9, 24,Z0001/	00016
DATA CJJS[1, 24.5], CUDS[2, 24.6], CUDS[3, 24.5]/ 9, 25.200DA/	91 000
DATA CDS(1. 25.6).CDDS(2. 25.6).CDDS(3. 25.6)/ 9. 26.ZQQEE/	91000
DATA CODS(1, 26.6), CODS(2, 26.6), CODS(3, 26.6)/ 9, 27,700FF/	00016
DATA QUS(1, 27.6).CDS(2, 27.6).CDS(3, 27.6)/10, 28.ZO187/	00016
DATA CDS(1. 28.6).CDS(2. 28.6).CDS(3. 28.6)/10. 29.Z0186/	00016
DATA CDS(1, 30,6),CDDS(2, 30,6),CDDS(3, 30,6)/10, 31,Z0180/	00016
DATA CODS(1, 31.6).CODS(2, 31.6).CODS(3, 31.6)/10, 32,Z0181/	00016
DATA COSTI, 32.6),CDDS(2. 32.6),CDS(3. 32.6)/11. 34.20001/	00016
DATA CDDS(1, 33.6),CDDS(2, 33.6),CDDS(3, 33.6)/ 7, 18,Z0035/	00016
F N D	- 00016
SUBROUTINE SERMES (PELBUF. OTBUF. PELMAX, VRES. SERCOT)	00016
c .	00016
IMPLIE IT INTESCRIA-2)	
REAL ESF	00016
C++++++ LABELEO COMMON /G3281T/ ******	00016
	9 1000
CCMMCN /G3281T/MASK(32).CMMASK(32).L18IT(32)Z8IT(32)	00016
INTEGER WASK COWASK LIBIT LIZRIT	00016
INTEGER WASK COWASK OLIBIT LIZBLY	00016
INTEGER WASK COWASK LIBIT LIZRIT	00016 00016 00016
C C C C C C C C C C C C C C C C C C C	00016 00016 00016
CCPMCN/FILES/TERM.LPFIL.PELFIL.OTFIL.ERFIL	00016 00016 00016 00016
C COMENTILE STERM LEFIL OFFIL ERFIL	00016 00016 00016 00016 00016
C CCPMCN/FILES/TERM.LPFIL.PELFIL.OTFIL.ERFIL C C C C C C C C C C C C C C C C C C C	00016 00016 00016 00016 00016 00016
COMMON ADGIC / SEARCH DIAG	00016 00016 00016 00016 00016 00016
COMMON LOGICAL SEARCH-DIAG	00016 00016 00016 00016 00016 00016 00016
COMMON ADGIC/SEARCH.DIAG CDMCALSEARCH.DIAG COMMON ASSOCIATION OF THE DEFINITION OF	00016 00016 00016 00016 00016 00016 00016
COMMON LOGICAL SEARCH-DIAG	00016 00016 00016 00016 00016 00016 00016 00016
INTEGER WASK COWASK LIBIT LIBIT CONTINUES CONT	00016 00016 00016 00016 00016 00016 00016 00016 00016
INTEGER WASK-COWASK-LIBIT-LZBIY C C++++++++++++++++++++++++++++++++++	00016 00016 00016 00016 00016 00016 00016 00016 00016
INTEGER WASK COWASK LIBIT LZBIY C C C C C C C C C C C C C C C C C C C	00016 00016 00016 00016 00016 00016 00016 00016 1***00016
INTEGER WASK COWASK LIBIT LIBIT. COMMON/FILES/TERM LPFIL PELFIL OTFIL ERFIL COMMON ADGIC/SEARCH DIAG LOGICAL SEARCH DIAG COMMON ADGIC/SEARCH BEGIN PROGRAM ************************************	00016 00016 00016 00016 00016 00016 00016 00016 00016 00016 00017
INTEGER WASK COWASK LIBIT LZBIY C C++++++++++++++++++++++++++++++++++	00016 00016 00016 00016 00016 00016 00016 00016 00016 00016 00017 00017
INTEGER WASK.COMASK.LIBIT.LZBIT C C C C C C C C C C C C C C C C C C C	00016 00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017
INTEGER WASK.COWASK.LIBIT.LZBIY C C C C C C C C C C C C C C C C C C C	00016 00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017
INTEGER WASK COWASK LIBIT LZBIY C C++++++++++++++++++++++++++++++++++	00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017
INTEGER WASK COWASK LIBIT LZBIY C ***********************************	00016 00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017
INTEGER WASK, COWASK, LIBIT, LZBIY C C++++++++++++++++++++++++++++++++++	00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017
INTEGER WASK, COMASK, LIBIT, LZBIY C ***********************************	00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017
INTEGER WASK COMASK LIBIT LZBIY C ***********************************	00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017
INTEGER WASK COMASK LIBIT LZBIY C ***********************************	00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017
INTEGER WASK, COMASK, LIBIT, LZBIY C ***********************************	00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017
INTEGER WASK COMASK LIBIT LIBIT COMMON TO THE COMMON ASK COMMON TO THE STERM LIBIT COMMON TO THE STERM LIBIT COMMON AS THE STERM LIBIT COMMON AS THE STERM COMMON AS T	00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017
INTEGER WASK-COMASK-LIBIT-LZRIY C ***********************************	00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017
INTEGER MASK COMASK LIBITALZRIY C C C C C C C C C C C C C C C C C C C	00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017
CCPMCN/FILES/TERM.LPFIL.PELFIL.GTFIL.ERFIL CCPMCN/FILES/TERM.LPFIL.PELFIL.GTFIL.ERFIL CDMMON/LOGIC/SEARCH.DIAG COMMON/LOGIC/SEARCH.DIAG COMMON/LOGIC/SEARCH.DIAG CCPMCN/FILES/TERM.LPFIL.PELFIL.GTFIL.ERFIL CCPMCN/FILES/TERM.LPFIL.PELFIL.GTFIL REWIND DELFIL REWIND DELFIL ERFOR=0 OTEL W=(PSILMAX+32-1)/32 CTINCT=0 CREAD AN ERROR FREE LINE 100 CONTINUE READ(1.END=600.ERR=800) INLNNO.INELCT.PELBUF IF (NCO(INLNNO-I.VRES).NE.0) GD TO 100 CREAD AN ERROR-CORRUPTED LINE 200 CONTINUE READ(2.END=500.ERR=600) OTLNNO.OTELCT.DTOUF CTLNCT=OTLNCT+IL	00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017
INTEGER MASK COMASK LIBITALZRIY C C C C C C C C C C C C C C C C C C C	00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017
INTEGER MASK.COMASK.LIBIT.LZBIT C***********************************	00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017
CCPMCN/FILES/TERM.LPFIL.PELFIL.GTFIL.ERFIL CCPMCN/FILES/TERM.LPFIL.PELFIL.GTFIL.ERFIL CDMMON/LOGIC/SEARCH.DIAG COMMON/LOGIC/SEARCH.DIAG COMMON/LOGIC/SEARCH.DIAG CCPMCN/FILES/TERM.LPFIL.PELFIL.GTFIL.ERFIL CCPMCN/FILES/TERM.LPFIL.PELFIL.GTFIL REWIND DELFIL REWIND DELFIL ERFOR=0 OTEL W=(PSILMAX+32-1)/32 CTINCT=0 CREAD AN ERROR FREE LINE 100 CONTINUE READ(1.END=600.ERR=800) INLNNO.INELCT.PELBUF IF (NCO(INLNNO-I.VRES).NE.0) GD TO 100 CREAD AN ERROR-CORRUPTED LINE 200 CONTINUE READ(2.END=500.ERR=600) OTLNNO.OTELCT.DTOUF CTLNCT=OTLNCT+IL	00016 00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017
INTEGER MASK.COMASK.LIBIT.LZBIT C***********************************	00016 00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017
INTEGER MASK.COMASK.LIBIT.LZBIT C***********************************	00016 00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017
INTEGER WASK.COMASK.LIBIT.LZRIT C ***********************************	00016 00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017
INTEGER MASK.COMASK.LIBIT.LZBIT C ***********************************	00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017
INTEGER MASK.COMASK.LIBIT.LZBIT C ***********************************	00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017
CCPMCN/FILES/TERM.LPFIL.PELFIL.OTFIL.EFFIL CCPMCN/FILES/TERM.LPFIL.PELFIL.OTFIL.EFFIL CDIPENSION PELBUF(60). DIBUF(60) COMMON ADGIC/SEARCH.DIAG LOGICAL SEARCH.DIAG CCOOMMON ADGIC/SEARCH.DIAG LOGICAL SEARCH.DIAG CCOOMMON ADGIC/SEARCH.DIAG CCOOMMON	00016 00016 00016 00016 00016 00016 00016 00016 00017
INTEGER MASK.COMASK.LIBIT.LZBIT C COMMON/FILES/TERM.LPFIL.PELFIL.OTFIL.ERFIL DIFENSION PELBUF(60). DYBUF(60) COMMON/LOGIC/SEARCH.DIAG LOGICAL SFARCH.DIAG LOGICAL SFARCH.DIAG REWIND PELFIL REWIND PELFIL ERFORD OTEL = (PILMAX+32-1)/32 CTLNCT=0 C READ AN ERROR FREE LINE 100 CONTINUE READ(1.END=600.ERR=800) INLNNO.INELCT.PELBUF IF (MCD(INLNNO-I.VRES).NE.0) GD TO TOO C READ AN ERROR-CORRUPTED LINE 200 CONTINUE READ(2.END=500.ERR=800) OTENNO.OTELCT.DEUF OTLNCT=JTLNCT+1 300 CONTINUE C COUNT DIFFERENCES BETWEEN TRANSMITTED AND RECEIVED LINES DO 450 I= 1.0TELW IF (UNDT.)INCT.PELBUF(1) GO TO 450 IF (UNDT.)INCT.DEUPELBUF(1) OTBUF(1) WR ITE(6.40) INLNND.OTENNO.I.PELBUF(1).OTBUF(1) 410 FORMAT (310.2212)	00016 00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017
TNIEGER MASK.COMASK.LIBIT.LZBIT C COMMENTILES/TERM.LPFIL.PELFIL.OTFIL.ERFIL C CUMENTILES/TERM.LPFIL.PELFIL.OTFIL.ERFIL C COMMON A.DGIC/SEARCH.DIAG LOGICAL SEARCH.DIAG LOGICAL SEARCH.DIAG C COMMON A.DGIC/SEARCH.DIAG LOGICAL SEARCH.DIAG C COMMON A.DGIC/SEARCH.DIAG REWIND PELFIL REWIND PELFIL ERFOR=0 OTEL W= (PILMAX+32-1)/32 CILNCT=0 C READ AN ERROR FREE LINE 100 CONTINUE READ(1.5END=600.5ER=800) INLNNO.INELCT.PELBUF IF (MCDINLNNO-I.VRES).NE.0) GD TO 100 C READ AN ERROR-CORRUPIED LINE C COUNT DIFFERENCES BETWEEN TRANSMITTED AND RECEIVED LINES DO 450 I=1.OTELW IF (OTBUFII).SEQ.PELBUF(I)) GD TO 450 IF (VNOT-21AG) GO TO 420 WRITE(6.410) INLNND.OTLNND.I.PELBUF(I).OTBUF(I) 410 FORMAT (318.2212)	00016 00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017
INTEGER MASK.COMASK.LIBIT.LZBIT C COMMON/FILES/TERM.LPFIL.PELFIL.OTFIL.ERFIL DIFENSION PELBUF(60). DYBUF(60) COMMON/LOGIC/SEARCH.DIAG LOGICAL SFARCH.DIAG LOGICAL SFARCH.DIAG REWIND PELFIL REWIND PELFIL ERFORD OTEL = (PILMAX+32-1)/32 CTLNCT=0 C READ AN ERROR FREE LINE 100 CONTINUE READ(1.END=600.ERR=800) INLNNO.INELCT.PELBUF IF (MCD(INLNNO-I.VRES).NE.0) GD TO TOO C READ AN ERROR-CORRUPTED LINE 200 CONTINUE READ(2.END=500.ERR=800) OTENNO.OTELCT.DEUF OTLNCT=JTLNCT+1 300 CONTINUE C COUNT DIFFERENCES BETWEEN TRANSMITTED AND RECEIVED LINES DO 450 I= 1.0TELW IF (UNDT.)INCT.PELBUF(1) GO TO 450 IF (UNDT.)INCT.DEUPELBUF(1) OTBUF(1) WR ITE(6.40) INLNND.OTENNO.I.PELBUF(1).OTBUF(1) 410 FORMAT (310.2212)	00016 00016 00016 00016 00016 00016 00016 00016 00016 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017 00017

UNCLASSI FI EU

IF (148()) BUF (1).J.1).NE.148(PELBUF(1).J.1)) ERRUR=ERRJR+1 440 CONTINUE 450 CONTINUE	0001
IF (OTLINU-I NLNNO) 200,100,580.	00017
C ERROR LINE NUMBER GREATER THAN GOOD LINE NUMBER:	00017
C COUNT DIFFERENCES BETWEEN GOOD AND ALL WHITE LINE	00017
SOO CONTINCE	0001
00 550 I= 1, OTELW	00017
IF (PE) RIF (1) - FO-O1 GD TO 550	0001
IF (.NOT.) IAG) GO TO 520	00017
WRITE(6.410) INLAND.OTLAND.I.PELBUF(I).OTBUF(I)	0001
CO 540 J= 1, 32	99917
IF (I48 (PELBUF(I).J.1).NE.O) ERROR=ERROR+1	00017
550 CONTINUE	00017
	00017
S80 READ(1.END=600.ERR=800) INLNND.INELCT.PELBUF IF(MCD(INLNNO-1.VRES).NE.0) GD TO 580	00017
	00017
GO TO 300	00017
CALCULATE ERROR SENSITIVITY FACTOR	00017
CALCOLATE BANGA SENSITIVITY FACILIE	00017
600 CONTINUE	00017
ESF=0. If (ERRCNT.LE.O) GO TO 650	00017
ESF-FLOAT (ERROR) /FLOAT (ERRENT)	00017
650 CONTINUE	00017
WRITE(6,700) ERRUR, ERR CNY, ESF, DYLNCY	00017
700 FORMAT('ONUMBER OF INCCRRECT PELS = 110/	0001
* ** ** ** ONUMBER OF RITS IN FRROR TRANSMITTED #* ** 110/	00017
* 'OERROR SENSITIVITY FACTOR =', F12.4/	0001
* "O TO TAL NUMBER OF DUTPUT LINES PROCESSED = ".18)	00017
RE TURN	90017
STEP 800	00017
E N D	0001
SUBROUTINE STATS(LENGTH-INLNCT-DIAG)	00017
IMPLICIT INTEGER(A-Z)	0001
INTEGER ATT (5) ASTT (245) ALENGTH (SMLNCT)	00017
REAL STT(2,5),SUM,SUMSQ	90017
LOGICAL DIAG	00017
LIFE DELINITION2 44444444444444	00017 00017
CCMMCN/FILES/TERM.LPFIL.PFIL.OTFIL.ERFIL	00017
CATA MTT/0-24-AB-06-102/	00017
CATA MTT/0, 24, 48, 96, 19 2/	90017
************************	*****00017
DC 3C0 I=1.5	0001 7
ITT(1, 1)=10000	00017
	00017
SUM=0. SUMS Q=0.	00017
	00017
	00017
FIND FILLED LINE LENGTH	00017
LEN=MAXO(LENGTH(J).MTT(I))	00017
IF (DIAG) WRITE(6.50) LEN	00 01 7 00 01 7
50 FORMAT(18)	00017
FIND PINIMUM LINE LENGTH	00017
	90017
ITT(1.1)= MINO (LEN. ITT (1.1))	00017
FIND MAXIMUM LINE LENGTH	00017
	00 01 8 00 01 8
ITT(2. 1) = MA XO(LEN. ITT(2.1))	00018
	8 1000
	00018
SUM=SUM+FLOAT (LEN)	00018
100 CONTINUE	61000
PIND SUM OF LENGTHS BUM=SUM+FLOAT(LEN) SUM SUM SUM SUM SUM FLOAT(LEN)) ##2 100 CONT INUE	000 000 000 000

5	FI	ND	SAMPLE	MEAN	AND	STAN	DARD	DEVIA	TION						00018
ئے															000i A
		ST	[[] • []=	SUM/FI	LOAT	INFF	CT)								00018
			r(2.1)= NTINUE	SURTE	(SUMS	94-(5	UNTT	2)/FLO	ITCIN	LNC	11)//	.DATL	INTINC	T-1))	00018
C	300	COI	111105												99918
		WR:	ITE(6.4	00)(1	TT(1.	1).1	=1.5)							00018
_	400	FOI	TAPE												00018
		:0	C 00 50				N	IINIMU	4 TRA	NSM	I 551 D1	ITIM	E (48	00 BPS) .	//00018
			COPO L ENS					MS	5 M	is.	10 MS	2	O MS	AO MEI	00018
	-	i	STATIS		•//		•	, 43	3 "		10 113		y ma	40 H3 1	00018
_	+		ME NE 4	W-11	ONVS (
			ITE(6.4	10)(1,	TT(2.	11.1	= 1.51)							00018
_		FU	FYAT (UNITED I	NY - W	1817	77-								00018
			ITE(6.)						•	00018
_	A20	EO.	TAKE.												000 18
		•		E JEA									•		00018
			ITE (6.4	30)(5	11(2	17.1	=1.5	<u> </u>							00018
		•		ARD C	FV IA1	TON	. 2X.	5(F8.2	1 1						00018
C									•						00018
			TURN												00018
^		E	N D	05 00	ec		000	-0.44				OCT		1829	00018
4			ENU	! P	<u> </u>	PIXI	PRIM	KAM			LINE	_ KPT	MEUE	1829	
_															
_															
_															
_															
_								·							
_				· ———											
_															
_				· · · · · · · · · · · · · · · · · · ·											
_															
_															
_															
_															
	•														
_															
-															
														•	
_															
_															